Convolutional Neural Networks - Generalizability and Interpretations

Malmgren-Hansen, David

Publication date:
2018

Document Version
Publisher's PDF, also known as Version of record

Convolutional Neural Networks - Generalizability and Interpretations

David Malmgren-Hansen

Kongens Lyngby 2017

Technical University of Denmark
Department of Applied Mathematics and Computer Science
Richard Petersens Plads, building 324,
2800 Kongens Lyngby, Denmark
Phone +45 4525 3031
compute@compute.dtu.dk
www.compute.dtu.dk
PHD-2017-459, ISSN: 0909-3192

Summary (English)

Sufficient data is key when training Machine Learning algorithms in order to obtain models that generalize for operational use. Sometimes sufficient data is infeasible to obtain and this prevents the use of Machine Learning in many applications. The goal of this thesis is to gain insights and learn from data despite it being limited in amount or context representation. Within Machine Learning this thesis focuses on Convolutional Neural Networks for Computer Vision. The research aims to answer how to explore a model's generalizability to the whole population of data samples and how to interpret the model's functionality. The thesis presents three overall approaches to gaining insights on generalizability and interpretation. First, one can change the main objective of a problem to study expected insufficiencies and based on this make better a choice of model. For this first approach the thesis presents both a study on translational invariance as well as an example of changing the objective of a problem from classification to segmentation to robustly extract lower level information. The second approach is the use of simulated data which can help by inferring knowledge in our model if real data is scarce. The results show clear advantages both when using rendered Synthetic Aperture Radar images, but also when predictions from physical models are used as target variables which are matched with real data to form a large dataset. The third approach to cope with data insufficiencies is to visualize and understand the internal representations of a model. This approach is explored and concrete examples of learnings that can be obtained are shown. There is no doubt that large quantities of well representing data is the best foundation for training Machine Learning models. On the other hand, there are many tools and techniques available to interpret and understand properties of our models. With these at hand we can still learn about our models and use this knowledge to e.g. collect better datasets or im-

prove on the modeling.

Summary (Danish)

Tilstrækkelige mængder af kvalitets data er vigtigt til at træne Machine Learning algoritmer der generaliserer til operationelt brug. I visse tilfælde er det dog ikke muligt at indsamle nok data og det forhindrer brugen af Machine Learning til mange problemer. Målet for denne afhandling er at opnå viden fra data selvom data er begrænset i mængde eller i den kontekst det beskriver. Indenfor Machine Learning fokuserer denne afhandling på Convolutional Neural Networks i billed analyse. Forskningen stræber efter at besvare hvordan vi kan sikre at vores datamodeller generaliserer til hele populationen af data samt hvordan vi fortolker vores modellers funktionalitet. Afhandlingen præsenterer tre generelle indgangsvinkler til at opnå indsigt om generalisering og fortolkning. Den første indgangsvinkel studerer både variation i objektplacering i billeder samt hvordan man kan skifte fra en klassifikationsmetode til segmentering når man vil vurdere forskellige algoritmer. Den anden indgangsvinkel drejer sig om brugen af simuleret data som hjælper ved at overføre den viden vi allerede har om problemets natur til et Convolutional Neural Network når rigtig data er knap. Resultaterne viser klare fordele både når man bruger syntetiserede billeder i træningen af en model, men også når forudsigelser fra fysiske modeller bruges som output variable der matches med rigtig data. Den tredje indgangsvinkel er at lære om vores datamodeller ved at visualisere deres repræsentation af data. Denne indgangsvinkel udforskes og konkrete eksempler på hvad man kan lære vises i afhandlingen. Der er ingen tvivl om at store mængder af struktureret data er det bedste fundament for Machine Learning, men der findes mange teknikker og metoder til at fortolke og forstå egenskaberne af vores modeller. Med disse metoder kan vi stadig lære om vores modeller og bruge den viden til at forbedre dem ved f.eks. at målrette indsamlingen af data eller forbedre modellernes egenskaber.

Preface

This thesis is part of the fulfillment of the Danish Industrial PhD program. The project is a collaboration between the Technical University of Denmark and Terma A/S with support from the Innovation Fund Denmark.

The PhD project entitled "Classification of Targets in Synthetic Aperture Radar Imaging" set out to investigate classification models for Synthetic Aperture Radar images. While the title and scope of this thesis have changed slightly to focus on the statistically derived models known as Convolutional Neural Networks, the thesis still answers essential questions for the project. Since Convolutional Neural Networks have in recent years been considered state of the art models in image classification, segmentation, etc., these constituted a promising solution to the problem at hand. In order to mature these models for the application, a deeper understanding was needed. Often, we face the problem that fully representative datasets are hard to obtain. How we can gain information from the data we have and how we can increase model generalization is the scope of this thesis. The thesis presents different approaches to obtain insights when dealing with datasets of limited representativity. These topics are ordered in three parts, Experimental Setup, Simulated Data and Interpretations.

Lyngby, 31-August-2017

David Malmgren-Hansen

Included Papers

A Malmgren-Hansen, D., Engholm, R., & Pedersen, M. O. (2016, June). Training Convolutional Neural Networks for Translational Invariance on SAR ATR. In *Proceedings of EUSAR 2016: 11th European Conference on Synthetic Aperture Radar* (pp. 1-4). VDE.

B Malmgren-Hansen, D., & Nobel-J, M. (2015, December). Convolutional neural networks for SAR image segmentation. In *2015 IEEE International Symposium on Signal Processing and Information Technology (ISSPIT)* (pp. 231-236). IEEE.

C Malmgren-Hansen, D., Kusk, A., Dall, J., A. Nielsen, A., Engholm, R., & Skriver, H. (2017). Improving SAR Automatic Target Recognition Models with Transfer Learning from Simulated Data. *IEEE Geoscience and Remote Sensing Letters*.

D Malmgren-Hansen, D., Laparra, V., Nielsen, A. A., & Camps-Valls, G. (2017). Spatial noise-aware temperature retrieval from infrared sounder data. Presented at *2017 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*.

E Malmgren-Hansen, D., Laparra, V., Nielsen, A. A., & Camps-Valls, G. (2017). Statistical Retrieval of Atmospheric Profiles with Deep Convolutional Neural Networks. Submitted. **[under review]**

F Malmgren-Hansen, D., A. Nielsen, A. & Engholm R. (2017). Compressed Feature Visualizations in Convolutional Neural Networks. Submitted to *Neural Computing and Applications, Springer*. **[under review]**

Excluded Papers

- Nobel-Jørgensen, M., Malmgren-Hansen, D., Bærentzen, J. A., Sigmund, O., & Aage, N. (2016). Improving topology optimization intuition through games. In *Structural and Multidisciplinary Optimization, Springer* 54(4), 775-781.

Acknowledgments

I have had the pleasure of being guided by five supervisors during my time as a PhD student, Allan A. Nielsen, Rasmus Engholm, Rasmus Larsen, Henning Skriver and Morten Østergaard Pedersen, and I would like to thank them all for their contributions to my academic education and the research we performed. Allan A. Nielsen and Rasmus Engholm deserve a special thanks for the regular scientific discussions they patiently had with me. These discussions inspired my curiosity and educated me in my scientific approaches.

Besides my supervisors I have been fortunate to work with Gustau Camps-Valls, Valero Laparra, Anders Kusk, Jørgen Dall and Morten Nobel-Jørgensen, on projects that all resulted in scientific publications and I am deeply grateful for this and for the learnings it gave me. Gustau, Valero and the rest of the ISP group from University of València made my external stay with them a memorable experience. Not a day goes by without me missing the Spanish lifestyle, food and all the great people I spent time with during my stay.

The PhD program is a rewarding yet tough journey that matures your academic skills and provides you with a toolbox of scientific methods. The ability to throw yourself into demanding projects without a guarantee of any useful outcome requires a strong determined mentality. I will take this ability with me in my future career and always let the passion for knowledge influence my decisions.

Contents

Chapter 1

Introduction

The field of Computer Vision concerns a wide range of applications and problems related to automatic interpretation of visual data. The tasks can be to classify objects, segment images, render visual scenes, reconstruct geometries, measure color or structure, etc. Many of these problems have traditionally been solved by engineered image transformations that infer prior knowledge on the problem at hand. The advantage of this approach is that if we find improvements with the transformations, we know that our assumptions on prior knowledge were right. For example, in order to recognize the same object in two images taken from different positions, angles and illumination conditions, our representation needs to be scale, rotation and intensity invariant, as shown in [Low99]. Convolutional Neural Networks (Convnets) turn this approach around and aim to learn useful image transformations from a set of images. In this way we can solve problems without inferring prior knowledge and possibly explore trained models to learn about new relevant image transformations.

1.1 Scope

This thesis explains and explores the properties of Convnets that make them interesting in a Computer Vision context. The original project scope aimed to investigate current state of the art image classification algorithms. This goal has

been slightly changed by applying a focus on Convnets. A deeper understanding of Convnets was found necessary to progress towards practical solutions in Automatic Target Recognition (ATR) for Synthetic Aperture Radar (SAR) data. The overall goal is to explore how we can gain information from data in situations where our datasets have limitations. We refer throughout this thesis to limited datasets as datasets that are limited in either size, context representation (e.g. imbalanced categories) or datasets that lack the variances expected in the respective application (e.g. scale, rotation, or background variation). Limitations to datasets are critical to generalizability. In this thesis we think of generalizability as how well a model trained on a set of samples can be extended to the whole population. Closely connected with estimating generalizability is the ability to interpret a model's function. If we understand how a model works we have a better foundation for understanding the generalizability without testing the model on the whole population. The thesis goals are highly relevant for SAR ATR where high operational costs limit the possibility of collecting large suitable datasets. The findings from this thesis can be used for Convnets in general, however the experiments are mostly focused on Remote Sensing and SAR data, to remain in line with the original scope of the PhD project.

1.2 Outline

Chaper 2 - Theoretical Background, provides basic understanding of two topics which are important to the thesis, Convnets and SAR sensors. The following three chapters (3, 4, 5) covers the papers published during the PhD project as follows,

- Chapter 3, Experimental Setup - Paper A, B.
- Chapter 4, Simulated Data - Paper C, D, E.
- Chapter 5, Interpretations - Paper F.

Each chapter introduces the published contributions, explains their relevance in an overall context, and ends with a conclusion. Further, since not all work carried out in the project was published, some chapters contain additional experimental results. The thesis closes with a Discussion and Outlook, summarizing the findings and discussing the perspectives of the Convnet approach to solve computer vision problems.

Chapter 2

Theoretical Background

This chapter provides the reader with background information within two main topics of this thesis, Convolutional Neural Networks and Synthetic Aperture Radar data. If the reader is familiar with these subjects, this chapter can be skipped as it will not be referenced later in this thesis.

2.1 Convolutional Neural Networks

A Convnet is a Neural Network with one or more convolutional layers. The convolutional layer convolves a filter kernel over every variable position of an input as a opposed to a fully connected layer that typically would have a designated weight for each variable. The idea is to learn feature detectors, i.e. filters that enhance properties relevant to the given task while achieving invariance to irrelevant properties such as scale, rotation, translational shift etc. This is no different than what computer vision has aimed to achieve with manually engineered feature detectors in the past, but Convnets provide a framework to learn the features from data instead. Since a convolutional layer will look for local patters in input data (within the size of the kernel), rather than global patters as a fully connected layer, it is well suited for data that are, e.g. sampled with temporal or spatial relations. The local patterns found by a convolutional layer

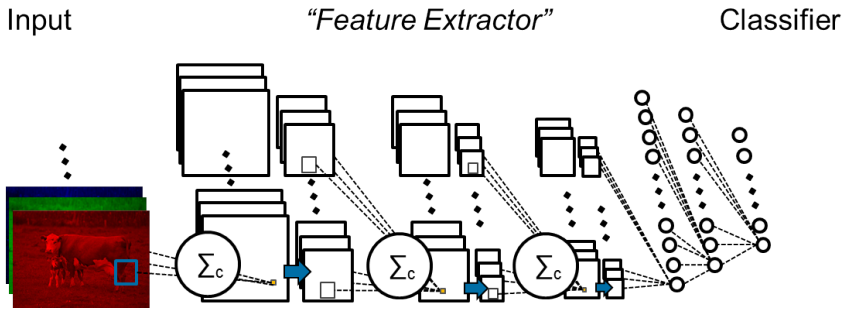


Figure 2.1: Example illustration of a Convnet from Paper F. Summation signs over c is sum over c input channels. Squares denote feature maps, i.e. outputs from convolutional layers. The blue arrows represent the chosen subsampling scheme e.g. max-pooling, and the small circles are neurons in fully connected layer

will become feature enhances/detectors for subsequent layers.

By stacking convolutional layers, see Figure 2.1, Convnets enable early layer filters to enhance simple features useful for describing a range of visual context classes, while deeper filters will become more specialized to solve the specific task. This hierarchical structure is information efficient in the sense of representing a range of image data context with as few features as possible. Whether the intended hierarchical structure is in fact learned is a challenge to verify, however visualization techniques for Convnet interpretations have shown it to be true for some datasets, [ZF14].

2.1.1 History

It is not the intention to give a thorough description of past research in Neural Networks. Four main events however summarize the contributions leading to the vast use of the models we are currently witnessing. These are the following:

1. McCulloch and Pitts (1943) introduction of mathematical models inspired by neural activities in the human brain [MP43]. Their work was followed up by Rosenblatt, [Ros58], who showed how a probabilistic version of these models could learn from observations.
2. The idea of learning internal representations by means of error-propagation [RHW85], known as back-propagation, was introduced by Rumelhart et al.

3. Yann LeCun et al. (1989), [LBD+89], applying back-propagation to Neural Networks that included convolutional layers inspired by studies on the visual cortex performed on animals by Hubel and Wiesel (1962) [HW62].
4. Alex Krizhevsky et al. performed a GPU accelerated implementation of a Convnet on a large scale image problem [KSH12].

Generally, the success of event 4 is largely credited to the availability of affordable fast computing platforms (GPUs) and access to large datasets compiled from internet downloads. Despite these facts playing a major role in their success, Alex Krizhesky et al. contributed with concepts that are now a part of the Neural Network research era from 2010 to 2015, in which the ease of Neural Network training was drastically improved. The most important contributions in this era are Rectified Linear Units (ReLU) [NH10], Dropout regularization [SHK+14], theoretically derived weight initialization [GB10], batch normalization [IS15] and data augmentation [KSH12].

Rectified Linear Units as activation functions, defined as $\sigma_{relu}(x) = max(0, x)$, have the property of a very simple gradient. The idea is that this leads to numerical stability and faster convergence during the training process, which was experimentally shown in [KSH12]. Further, one can achieve faster convergence by initializing weights in such a way that the product sum of inputs and weights for each layer sums up to approximately one, [GB10]. One could argue that initialization is redundant since the network should learn some solution despite initial values of weights, but due to the iterative update scheme of weights, clever initialization can lead to saving a large amount of computation. Since modern Convnets and computer vision tasks are large, saving computations can be the difference between converging within feasible time or not, and is therefore crucial. Further, since our update schemes are based on iterating towards an error minimum with a fixed or adaptive step size the initialization can influence whether we get stuck in a poor minimum or not.

For each solution to the weight matrix of a Neural Network there exists an infinite amount of equal solutions where weights upon following layers are scaled relatively. In principle we do not care which of these solutions we find, but we would like the weights not to shift between these relative scaling of weights during training. Often referred to as covariate shift, the relative transforming of weights between layers slows down the training process if the weights alter between these solutions. Batch Normalization aims to prevent this by always scaling the output of the network layers to have zero mean and unit variance. Such normalization reduces the Convnet learning capability, so batch normalization introduces two additional parameters to shift layer output distributions away from zero mean and unit variance. These two parameters are updated during gradient descent training together with the layer weights and can cancel the

normalization if necessary. Though it intuitively might seem contradicting that the normalization can be canceled, Batch Normalization has experimentally shown useful as a sort of regularizing constraint that enables higher learning rates, [IS15].

Dropout is another regularizing scheme. It aims to emulate ensemble model prediction of models with shared weights. The need for this arises since a real ensemble of Neural Networks would often be infeasible due to computational costs. Applied to a network layer, Dropout skips the update of each layer node with a given probability in an iteration of the update scheme. A node is either a neuron in a fully connected layer or a filter kernel from a Convolutional layer. Dropout prevents co-adaptation of weights and thereby over-fitting behaviours otherwise known as a common problem for Neural Networks. To further prevent over-fitting Data-augmentation can as well be applied. It aims to prevent over-fitting by considering every transformation of an input sample $\mathbf{x}$ that has relevance to the task being solved, as an additional training sample. Typical Data-augmentation transformations for images is zooming, rotation, scaling, shifting etc.

Another technique for Neural Network training that has eased the procedure is the adaptive control of learning rates. In order to achieve fast convergence and a stable reduction of the error function simultaneously, learning rates must be reduced during training. One can do this by linear reduction of the learning rate as a function of iterations in the training. Alternatively, two popular approaches named RMSprop, [HSS12], or ADAM, [KB14], can be used. These methods control learning rates with relation to the gradient of the error function.

2.1.2 Optimization

A set of weights for a Neural Network $\mathbf{W}$ is often found according to the maximum likelihood solution to the given problem. As we will see this yields a natural choice for the model's output and error function, [Bis06]. For a classification task with K mutually exclusive classes, Bayes theorem on the posterior probability of the k'th class can be rewritten,

$$\begin{aligned} P(C_k|\mathbf{x}) &= \frac{P(C_k)P(\mathbf{x}|C_k)}{P(\mathbf{x})} \\ &= \frac{P(C_k)P(\mathbf{x}|C_k)}{\sum_{j=1}^{K} P(C_j)P(\mathbf{x}|C_j)} \qquad (2.1) \\ &= \frac{e^{a_k}}{\sum_{j}^{K} e^{a_j}}, \quad a_k = \ln(P(C_k)P(\mathbf{x}|C_k)) \qquad (2.2) \end{aligned}$$

In Equation 2.1 the observation probability, $P(\mathbf{x})$, is expanded with the law of total probability. Equation 2.2 is known as the softmax function and when used as output function in a Neural Network, the k'th output a_k for $k = 1, ..., K$ will represent the logarithm of the evidence, $P(\mathbf{x}|C_k)$, multiplied with the prior probability on the class, $P(C_k)$. Given the probabilities from the network we shall consider an error function (also known as the objective or loss) to explain the distribution of an estimated target vector's probabilities. In most practical cases there is one class label per sample, which can be encoded as a vector $\mathbf{t}$ with one element equal 1 corresponding to the correct class and zeros in all other elements. For a mutually exclusive classification problem this is simply the product of individual Bernoulli distributed probabilities,

$$p(\mathbf{t}|\mathbf{x}, \mathbf{W}) = \prod_{k=1}^{K} y_k(\mathbf{x}, \mathbf{W})^{t_k} \tag{2.3}$$

Where $y_k()$ is the function for the k'th output of our Neural Network and t_k being the $k'th$ element of our target vector $\mathbf{t}$. When taking the negative logarithmic likelihood and considering the sum over all N samples in a given dataset we get an error function,

$$E(\mathbf{W}) = -\sum_{n=1}^{N}\sum_{k=1}^{K} t_k \ln(y_k(\mathbf{x}_n, \mathbf{W})) \tag{2.4}$$

Equation 2.4 is known as the categorical cross entropy.

In the case of a multi target regression with Neural Networks we follow the same approach exchanging the conditional distribution with the one appropriate for the given data. Commonly a Gaussian distribution is used and independent targets are assumed. With the independence assumption the likelihood is reduced to the product of the individual posterior probabilities of each target, Equation 2.5.

$$p(\mathbf{t}|\mathbf{x}_n, \mathbf{W}) = \prod_{l=1}^{L} p(t_l|\mathbf{x}_n, \mathbf{W}), \tag{2.5}$$

n represents a specific sample over and L is the number of target variables in our regression. The target $\mathbf{t}$ is no longer an encoding of class probabilities but a vector of continuous target variables with t_l being the l'th element. If we consider our prediction a target estimate with a Gaussian distributed error,

$$\mathbf{t} = \mathbf{y}(\mathbf{x}_n) + \mathbf{e}_n, \quad e \sim N(0, \sigma) \tag{2.6}$$

we can obtain a maximum likelihood solution to the regression,

$$p(t_l|\mathbf{x}_n) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{\frac{(y_l(\mathbf{x}_n, \mathbf{W}) - t_l)^2}{2\sigma^2}} \tag{2.7}$$

This reduces to the sum of squares error function when taking the negative log-likelihood and discarding constants. We can write this over all N samples in our dataset for all L targets as,

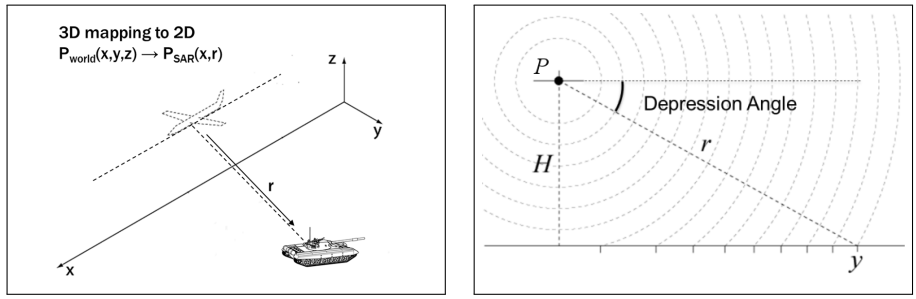
$$E = \sum_{n=1}^{N} \sum_{l=1}^{L} (y_l(\mathbf{x}_n, \mathbf{W}) - t_{l,n})^2 = \|\mathbf{y}(\mathbf{x}_n, \mathbf{W}) - \mathbf{t}_n\|^2 \tag{2.8}$$

which gives the least squares error function. The output function of the Neural Network should be selected appropriately. If we do not make assumptions on the nature of our signal, the output of the last layer in the model should be a linear projection of the previous layer's outputs. In some cases there might be reasons to change the output function, e.g. if we are predicting targets that have upper or lower bounds.

2.2 Synthetic Aperture Radar

A Synthetic Aperture Radar (SAR) is an imaging sensor based on the principles of radar technology. It has to be operated on a moving platform, typically an airplane or satellite, and it records the terrain in a line scanning manner with each line perpendicular to the flight trajectory. Since a SAR is based on emitting electromagnetic waves from a source carried on board the platform, it is capable of recording day and night. Further, since the SAR sensors operate at lower frequencies than optical systems they are capable of looking through clouds. The capabilities to record any time of day and in all weather conditions are very useful for many applications. Figure 2.2 depicts the geometry of a SAR system in a 3D coordinate system representing its environment. By having an antenna looking widely, the radar receives target reflections from several positions along its trajectory (x). The varying distance to the target within these reflections leaves a modulated pattern in the received signal that is given by distance to the target, the antenna beam width and the velocity of the SAR platform. When the signal is demodulated, energy is concentrated in the center position of the received signal, i.e. the SAR signal is focused.

There are three distinct features commonly present in SAR images. As a SAR system emits a coherent signal in order to measure the phase of reflections, an unwanted signal called speckle arises when multiple scatterers are present in one resolution cell. This is due to the complex summation of the multiple scatterers, and speckle will follow a Rayleigh distribution in the amplitude component. If $V = (X, Y)$ denotes a vector with two independent Gaussion distributed elements with zero mean and equal variance, the Rayleigh distribution is the



(a) (b)

Figure 2.2: (a): A SAR trajectory illustrating how a 3D coordinate system is mapped along trajectory axis, x, and slant range axis, r. (b): Depression angle illustration. P is the SAR sensor position, H the height above ground and y the ground range axis.

distribution of the length L given,

$$L = \sqrt{X^2 + Y^2} \tag{2.9}$$

A SAR signal is measured by two complex components, (x, y), in rectangular coordinates. Speckle will influence each component with a Gaussian distributed variance. So the joint distribution of (x, y) becomes,

$$(x, y) = x + iy \tag{2.10}$$

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{x^2}{2\sigma^2}} \tag{2.11}$$

$$p(x, y) = p(x)p(y) = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \tag{2.12}$$

When the complex signal is converted to polar coordinates to get the amplitude of the backscatters the amplitude distribution will be,

$$ae^{i\theta} = x + iy \tag{2.13}$$

$$a = \sqrt{x^2 + y^2} \quad \vee \quad \theta = tan^{-1}\frac{y}{x} \tag{2.14}$$

$$p(a) = \frac{a}{\sigma^2} e^{-\frac{a^2}{2\sigma^2}}, \quad a \geq 0 \tag{2.15}$$

$$p(\theta) = \frac{1}{2\pi}, \quad 0 \leq \theta < 2\pi \tag{2.16}$$

where we see that the amplitude follows a Rayleigh distribution while the phase is uniformly distributed. The relationship between Equation 2.12 and 2.15 can be found by exchanging variables and introducing an equal integral in rectangular and polar coordinates. Speckle is a multiplicative signal source hence higher

backscatter will lead to higher amount of speckle. Often SAR images are filtered with a spatial moving average operation (multilook SAR image) to reduce the speckle.

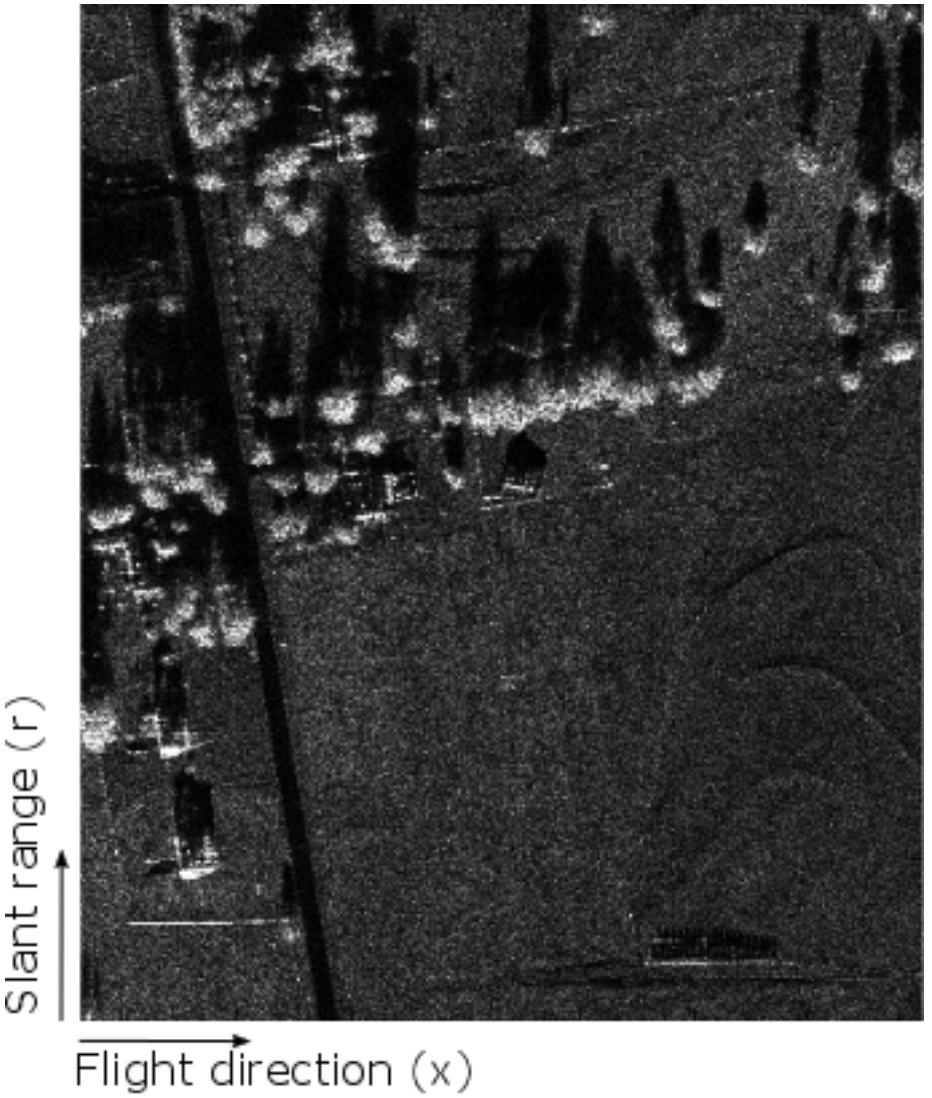


Figure 2.3: SAR terrain picture from the MSTAR project (background image).

Figure 2.3 shows a SAR image of a terrain, with a road (dark - i.e. low backscat-

ter), fields, trees and some houses/buildings. A common feature of SAR images can easily be observed in form of shadows behind each object rising above the ground. Due to the recording geometry of a SAR, objects that rise above ground will cast shadows behind them. Lower depression angles will yield longer shadow casts. Another feature commonly found in SAR images is foreshorting. Since signals are recorded along the r-axis on Figure 2.2 (slant range), reflections from elevated scatterers may appear before reflections from closer to the ground. This can make tall trees or mountains appear distorted compared to aerial recordings from optical instruments.

2.2.1 Polarimetric SAR

A number of configurations for SAR sensors need to be chosen during the design phase dependent on the application. Spotlight SAR are systems with steerable antennas that focus on a given area in order to achieve the higher cross range resolution at the cost of lower ground coverage. Different radar frequencies can be chosen depending on the application. Examples of applications where frequency is important are forest biomass estimations or estimation of water content in snow, [BFG+99, RHP+94]. In the application of forest biomass estimation certain SAR frequencies are suitable, as scattering is received from both tree crowns as well as the forest floor. Another SAR design choice is polarization of the transmitted and received radio waves. Fully polarimetric SAR systems can provide additional information about scatterers. The additional information comes from the interaction between polarized transmitted signal and the scatterer geometry. The orientation of different parts of an object's geometry will result in different polarized backscatter signatures. A fully polarimetric SAR acquires four backscatter measures at different combinations of transmitted and received linear polarizations, HH, HV, VH and VV. An average of the two signals from the cross polarized measurements S_{HV} and S_{VH} can be used to reduce the noise on these components, [Skr12]. A scattering covariance matrix, [UE90], can be constructed for every pixel as,

$$\mathbf{k} = [S_{HH} \quad S'_{HV} \quad S_{VV}]^T, \quad S'_{HV} = 0.5(S_{HV} + S_{VH})$$

$$\mathbf{Z} = \frac{1}{N}\sum_{n=1}^{N} \mathbf{k}(n)\mathbf{k}(n)^{*T}$$

$$= \begin{bmatrix} \langle |S_{HH}|^2 \rangle & \langle S_{HH} S'^{*}_{HV} \rangle & \langle S_{HH} S_{VV} \rangle \\ \langle S'_{HV} S^{*}_{HH} \rangle & \langle |S'_{HV}|^2 \rangle & \langle S'_{HV} S^{*}_{VV} \rangle \\ \langle S_{VV} S^{*}_{HH} \rangle & \langle S_{VV} S^{*}_{HV} \rangle & \langle |S_{VV}|^2 \rangle \end{bmatrix}$$

where $\langle . \rangle$ denotes the multilook operation and $*$ the complex conjugate. Since the covariance scattering matrix is hermitian, i.e. the off diagonal elements are

complex conjugates of each other, each pixel in our final polarimetric SAR image can be represented by the six unique elements. The off-diagonal elements are complex numbers while the diagonal elements are real numbers. Polarimetric SAR images can be visualized as RGB color images by their diagonal elements of the scattering covariance matrix. A common used color encoding is $|S_{HV}|^2$ for the red channel, $|S_{HH}|^2$ for the green and $|S_{VV}|^2$ for the blue, as this yields the most natural looking colors. An example of this encoding can be seen for the EMISAR dataset on Figure 2.4.

Figure 2.4: Color coded polarimetric SAR image from the EMISAR Foulum dataset, June recording.

Such color encodings can help visualize the differences of scattering patterns in the signal. In Figure 2.4 we see different colors on the fiels which arise from the

different polarimetric scattering signatures.

Chapter 3

Experimental Setup

In order to solve a classification problem, a Convnet needs a representative set of sampled training data which covers expected variances. In many sciences and applications, gathering sufficient data with associated targets values can be a very hard task and remote sensing is no exception. If we consider a reflectance spectrum measured by an airborne optical instrument, the reflectance will include variance due to time of the day, weather conditions, atmospheric conditions, view incidence angle, geographic location, transient changes, etc. One can tackle this data challenge in two ways. The first option is to collect as large a dataset as possible and aim to cover enough variance. While this is not always possible, another approach is to accept the limitations of gathered data and try to understand the model's capabilities and ability to generalize. With this understanding new data collection can be focused to extend the models capabilities or models with built-in invariance towards the data pitfalls might be explored. In this chapter we consider a simple classification task on a limited dataset. By changing the objective from studying the accuracy on this task, which we know do not generalize to operational use, we can still obtain learnings from the dataset.

3.1 Classification

The MSTAR dataset was recorded during the mid nineties in order to improve the data foundation for research in Automatic Target Recognition (ATR) for SAR images. It consists of 10 military vehicles recorded with an x-band SAR radar at 30x30 centimeter pixel resolution. A description of the dataset can be found in Appendix A.1. MSTAR is known to be a very limited dataset in regards to meeting the variability of an operational scenario and according to [RWV+98] it can not be considered a random subset of real world data. It lacks variability both in terms of depression angles where it is limited to 15° and 17° but also in terms of background variation and the very limited number of vehicles recorded. The dataset is not meant to give a performance score of operational SAR ATR, but can still be used to gain insight into how SAR data can be modeled. As the training set contains solely 17° depression angles and the test set 15° the model's ability to generalize over 2° depression angle variation can be studied. Since the depression angle is changing the appearance of objects in SAR images significantly due to the slant range and ground range relationship, robustness towards depression angle variation is relevant to study.

One of the first benchmarks of Convnets on MSTAR was performed in [Mor15]. The network used in [Mor15] is relatively shallow compared to the ones reported in [SVZ13] and [SLJ+15] for the ImageNet large scale image recognition challenge. Given the more simple problem in MSTAR and a lot fewer images it is reasonable to believe a smaller architecture is sufficient. When it comes to designing an architecture for a Convnet there is little theoretical foundation to rely on. Deep Learning is an experimentally driven field, and following best practices and related work is the best initial starting point for every new problem. Automatic hyperparameter optimization can at best be applied in a subset of the hyperparameter space since there is no limit to how many hyperparameters a Convnet architecture can have. For every new layer we add, we can change its size, its activation function, whether to include pooling layers or not, and even more. Further, when the number of parameters grow, so does our training time, and this always introduces an upper practical limit to how big models we can explore. While the Convnet reported in [Mor15] achieved a reasonable performance of 92.3% we found that it could be additionally improved by following some best practices developed before and after the publication. These build on the following concepts and experiences,

1. Logarithm transform of SAR pixel values. Since the MSTAR images contained some pixels with value 0, we added a small number, took the 10 base logarithm, before normalizing pixels to zero mean and unit variance. In SAR applications logarithmic transform is often performed in order to

make the multiplicative speckle signal additive. Further, the logarithmic transformation has advantages when visualizing SAR images as the high dynamic signal range is mapped so lower backscatter patterns are easier visible together with higher backscatters.

2. Smaller filter kernels impose parameter and computational sparsity, [SVZ13]
3. Dropout regularization. Though often resulting in a need for higher number of parameters in a model, dropout is often seen to increase the end-performance, [SHK+14].
4. Batch Normalization layers, [IS15], added after each convolutional layer in order to reduce covariate shift and thereby achieve faster convergence.
5. Adam (Adaptive Moment), [KB14], optimizer scheme to continuously control learning rates during optimization as opposed to the fixed learning rates in [Mor15] that were reduced by a factor of 10 after a 3000 iterations.

The final network architecture of our proposed network can be seen in Table 3.1. With this architecture we reach a performance of 99.19% accuracy after 1000 epochs, which takes $1\frac{1}{2}$ hour trained on a NVIDIA Titan Black GPU.

A benchmark of different classifiers on MSTAR was performed as an initial study in this project and can be seen on Figure 3.1. The SVD+SVM method refers to an approach inspired by [DS83], where 10,000 $9x9$ pixel patches were extracted from the training images in MSTAR and a singular value decomposition over these was performed. This lead to a set of decomposed patches explaining the majority of local variances in the images which can be used as feature extraction filters.

Since the Convnet and the SVD+SVM methods contain a filtering step before classifying the images, it is not surprising that they perform well. More surprising is the fact that some of the other approaches, that work entirely based on finding decision boundaries between the pixel vectors that the MSTAR images span. The linear Support Vector Machine (SVM), K Nearest Neighbours (KNN) and Linear Disciminant Analysis (LDA) all achieve $>75\%$ accuracy. Keeping in mind the unrealistic simplicity of MSTAR images misleading choices of classifiers can be made if only tested on MSTAR. Alternatively, one can study specific robustness of classifiers with simulated data.

In Paper A we show that Covnets perform better compared to other classifiers when introducing translational variance. The experiments were performed on a simulated SAR ATR dataset in order to create different datasets with different amount of object translation. By using a simulated dataset we ensure that

Table 3.1: The Convnet architecture presented below has 763678 trainable parameters. It reached a test accuracy of 99.2% on MSTAR 10-class benchmark. The process names refer to function names from the Deep Learning software library Keras, [C+15].

Process	Parameters	Parameters Description
Input	(128,128)	MSTAR image size [pixels]
Conv2D	(12,5,5)	(Number of kernels, kernel width and height)
BatchNormalization		
Activation	ReLU	Function type
MaxPooling2D	3x3	Pooling window size.
Dropout	30%	Probability of a node getting dropped.
Conv2D	(36,5,5)	(Number of kernels, kernel width and height)
BatchNormalization		
Activation	ReLU	Function type
MaxPooling2D	2x2	Pooling window size.
Dropout	30%	Probability of a node getting dropped.
Conv2D	(72,5,5)	(Number of kernels, kernel width and height)
BatchNormalization		
Activation	ReLU	Function type
MaxPooling2D	2x2	Pooling window size.
Dropout	30%	Probability of a node getting dropped.
Dense	144	Fully connected layer size.
BatchNormalization		
Activation	ReLU	Function type
Dropout	50%	Probability of a node getting dropped.
Dense	144	Fully connected layer size.
BatchNormalization		
Activation	ReLU	Function type
Dropout	50%	Probability of a node getting dropped.
Dense	10	Fully connected layer size.
Activation	softmax	

models are perfectly centered initially. It is shown that a random translation of the vehicle outside the image center by as little as 3 pixels drastically decreases the performance on the tested classifiers except for the Convnet, see Figure 3.2. Now, if we expect this kind of translational variance in real world data we know we should look for classifiers with similar properties of a Convnet (filtering/feature extraction combined with pooling schemes) when studying a SAR ATR application. The details of the models tested in Figure 3.2 and the simulated dataset used can be found in Paper A.

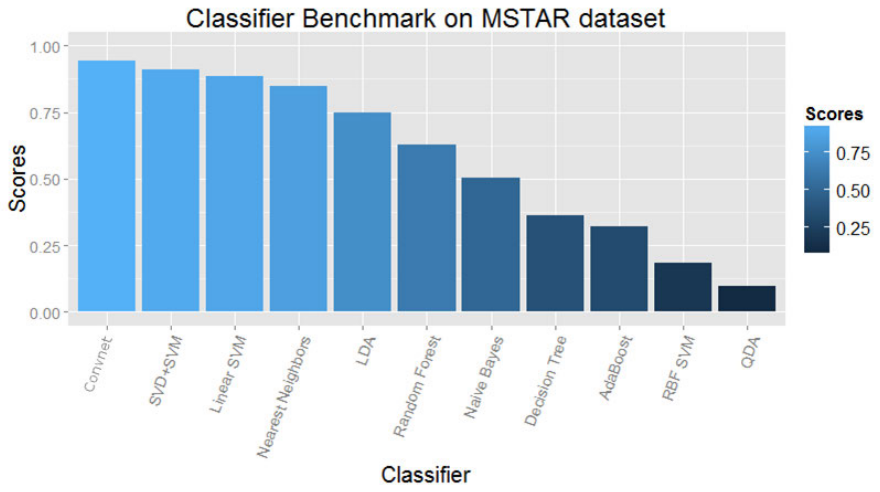


Figure 3.1: Benchmark of different classifiers on the MSTAR dataset. The Convnet performance shown here was before applying improvements described in Table 3.1.

3.2 Segmentation

Alternatively to applying specific tests like translation invariance to gain information from limited datasets like MSTAR we might consider new ways to extract information. An example could be to change the objective from classification to segmentation and reduce the semantic problem to background, target and shadow. This will enable us to pool the MSTAR data and create a labeled dataset in a new way. Due to the geometric properties of SAR, one can roughly estimate the object size with a good segmentation mask of a target and its shadow. SAR object segmentation can be performed with Convnets by considering it a pixel-wise classification. We propose to follow a method where each pixel is classified from a neighborhood of pixels to be either background, object or shadow. In Paper B we enable this by estimating the ground truth pixel annotation from Computer Aided Design (CAD) models of the targets in MSTAR. First the CAD model is converted to a depth map by computer graphic rasterization given the radar view angles. Secondly, these pixel-wise distances are mapped corresponding to SAR geometry. The pipeline of this approach is illustrated in Figure 3.3. This technique yields a simple annotation of every pixel in the image that can be used to train a supervised segmentation algorithm. Our annotation masks for all MSTAR images in the 10-way classification tasks are publicly available via [MHNJ17]. The Convnet approach performs

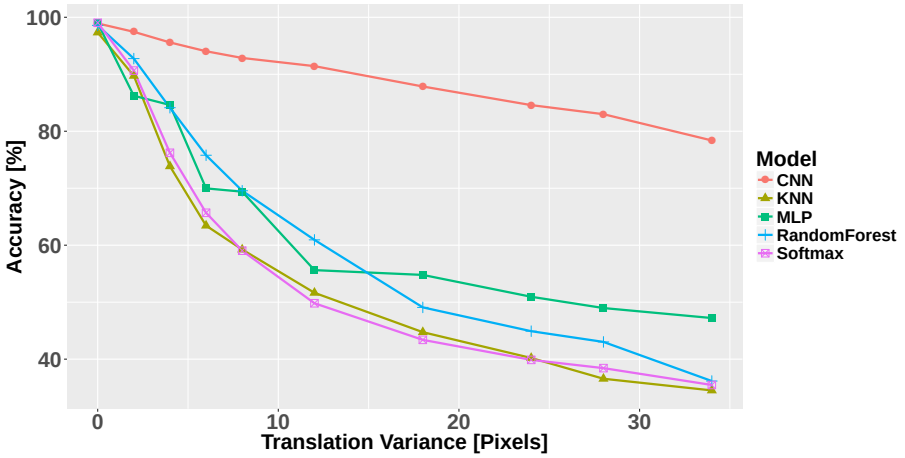


Figure 3.2: Translational invariance results of the five tested classifiers. Datasets of simulated SAR ATR data were generated with different amount of target off-center translation in the image. For each dataset the classifiers were retrained and tested. While the complexity of the task increases with translation, the dataset size remained the same which explains the small decrease in performance of the Convnet as well.

very well on the segmentation, specifically in finding the boundary between a target and its shadow. Other work on SAR segmentation algorithms typically classifies the boundary as background pixels due to the similarity of the values [WKC+99, AWZ02, HWH16]. However, finding the boundary is important to estimate the target height above ground.

3.2.1 The EMISAR Experiments

The pixel-wise classification approach can be adopted to other applications with even more diverse pixel classes. This is illustrated by applying the approach to the EMISAR Foulum, Denmark dataset of crop classification, [SST99]. The EMISAR is a fully polarimetric SAR radar, i.e. capable of both transmitting and receiving horizontally and vertically polarized electromagnetic waves. Further, EMISAR is a dual radar system with both L-band and C-band recording simultaneously with the goal of studying optimal configurations for crop classification. The SAR map over the fields of Foulum are shown in Figure 3.4 with the outlines of their label masks. Pixels left of the dashed line are held aside for the test set while the part on the right is used for training. The dataset

(a) (b) (c) (d)

Figure 3.3: Pipeline in the SARBake algorithm presented in Paper B. Given a MSTAR target vehicle (a) find a CAD model (b) render a depth map from the radar view position (c) and convert the depths into a mask of the target. Shadow distances can be calculated to a flat ground easily by triangulation from edge points, but in our approach a flat ground was rendered where the shadow distances can read from. This allows for different terrain models in more complex scenarios.

consists of polarimetric scattering covariance matrices in each pixel position. The off-diagonal elements are complex numbers and when these are stacked together with the diagonal elements as separate image channels, the resulting image size of the EMISAR Foulum dataset are 1024x1024x9. As a preprocessing step these channels are normalized by subtracting mean and dividing by standard deviation. The diagonal elements of the covariance matrix are Gamma distributed. From experiments it was found an advantage to take the logarithm of the Gamma distributed elements before performing a zero mean unit variance normalization. A patch of size 21×21 is extracted around every pixel containing an annotated field. To avoid overlap between patches in test and training sets, a ten pixel wide strip to the right of red line in Figure 3.4 is left out. The approach is to train a Convnet to classify the center pixel of each patch. To cope with an unbalanced amount of samples from the different crop types smaller classes where oversampled. This yielded better performance than under-sampling big classes or weighing the loss function with the inverse samples size. With a model architecture similar to the one used in Paper B, we achieved good performance compared with other work on this dataset, [Skr12], [VDLN12]. In [Skr12] it was shown that several temporal samples during a season of crop growth yield much higher accuracy when combined with the polarimetric and frequency information. Our approach achieved an error rate of 22% from a single temporal sample with all polarimetric and frequency information. [Skr12] used a much simpler approach to the classification of crop types and achieved 41% error rate from single temporal sample.

Table 3.2: Results from EMISAR experiments with different polarimetric and frequency settings. The performances are given in error rates over all test samples.

	L-Band	C-Band	L+C-Band
Diagonal polarimetric elements	33%	36%	25%
All polarimetric elements	34%	23%	22%

3.3 Conclusion

Given the deficiencies of MSTAR, the accuracy from a classification model on this dataset provides little knowledge about the problem of SAR ATR. By reformulating the problem into experiments that overcome known challenges, such as translation variance, we can still obtain useful information. Alternatively, we have shown how well performing segmentation algorithms can be trained on MSTAR and thereby yield information about object size in an automatic way. The segmentation might be an indirect way of gaining information about a vehicle, but it enables us to create an annotated dataset from existing data and train a robust algorithm on a reduced semantic content. The annotation masks have been made publicly available and provide a foundation for continuous experiments on MSTAR based on supervised segmentation models.

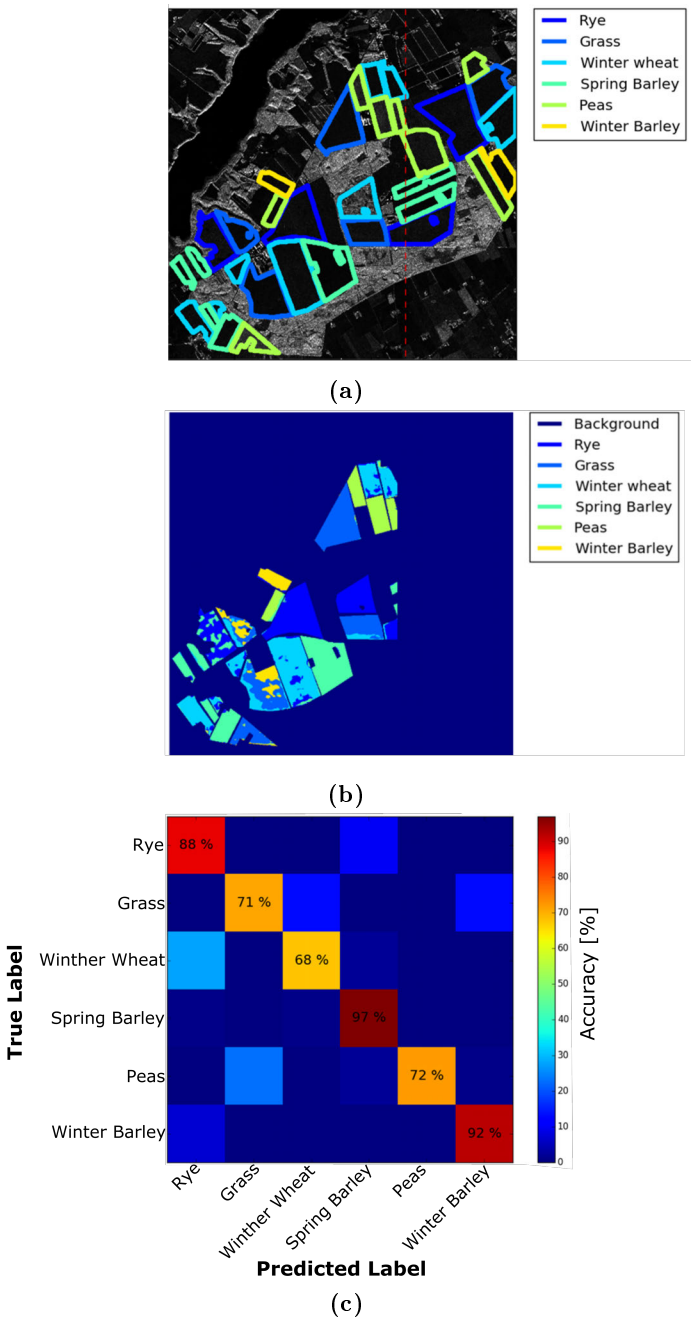


Figure 3.4: (a) EMISAR Foulum dataset with labels. (b) Convnet Prediction on test set. (c) Confusion Matrix for the 6 classes.

Chapter 4

Simulated Data

One of the success criteria for the Convnet by Alex Krizhevsky et. al [KSH12] in 2012 was the availability of the large, labeled image dataset, ImageNet, which has more than 1,000,000 images. Since 2012 Convnets have improved the score in many computer vision benchmarks. Often though, not by only training the Convnets on each benchmark dataset but largely also with help from the concept of Transfer Learning. Gathering a labeled set of more than a million images is a cumbersome task, but by first training a Convnet on ImageNet and then further training it on another computer vision dataset with adequate similarity, fewer images are needed in the second dataset. Transfer learning is still mostly applied between datasets with the same modality but the objects in the computer vision problem may be distinctively different, [She16, NGM15]. We will give an analysis of Transfer Learning by means of visualizations in Chapter 5. In this chapter we focus on how Transfer Learning can be used from simulated data and how simulated data in general can play a role, when real data is insufficient. Simulated data, being data generated from models of physical phenomena, can be seen as one way to add the existing prior knowledge when using Machine Learning models. In the traditional computer graphics context simulation are visual rendering of geometric models given knowledge of interaction between light and material. This type of simulation is great as an alternative to collecting datasets since it links the context to data samples and provides us with an automatic way of obtaining large quantities of data with target labels. For some applications, such as weather forecasting, large deterministic models

based on physical constraints creates the link between past observations and future predictions. These deterministic models can provide target variables for other models that link measurements from e.g. satellite sensors, to the observations e.g. temperature, wind speed, etc. which are needed for weather forecasting models to predict future states. This concept can be seen as another way where simulated data impose the physical knowledge we posses into Machine Learning models. In this chapter we will explore examples of both ways of generating datasets and explain how they can be useful when dealing with limited data.

4.1 Simulated Input Data

LeCun & Bottou released the NYU Object Recognition Benchmark (NORB) dataset in 2004, [LHB04]. NORB consists of stereo image pairs of ten toy figures from five generic object classes at several illumination levels and from different view angles. NORB can be seen as a way to simulate a real object recognition scenario, hence the generalizability from models trained on NORB is an interesting aspect. In [LHB04] encouraging results of the generalizability between NORB and real world objects were shown, although no training on images of real world objects was performed. Another way to gather data from a controlled set-up is by computer simulations. Due to the access to large databases like ImageNet and the concept of Transfer Learning, little need has been present for simulations of natural images. Simulation can be a part of the solution in computer vision problems, where transferring models trained on ImageNet do not provide a meaningful approach.

In Paper C we present the concept of Transfer Learning between simulated images and real images and provide an experimental analysis in the SAR ATR context. Recent advances in SAR simulation tools have enabled the use of simulated data as a part of the foundation for ATR models, [ØKCL16, KAD16]. An example from the SARSIM (Appendix A.2) dataset can be seen on Figure 4.1. It was developed by the National Space Institute at the Technical University of Denmark, [KAD16], and provides the foundation for the experiments in Paper C.

The main idea of Paper C is the concept of using Transfer Learning between simulated data, SARSIM, and the objective dataset, MSTAR. This concept will have several advantages over approaches that require simulation of the exact objects in the objective dataset. First, SAR ATR targets in real world applications are rarely known to a degree where precise CAD models can be obtained. Second, precise CAD models of certain objects are cumbersome to build and requires high amount of manual labor. With Transfer Learning we aim to learn

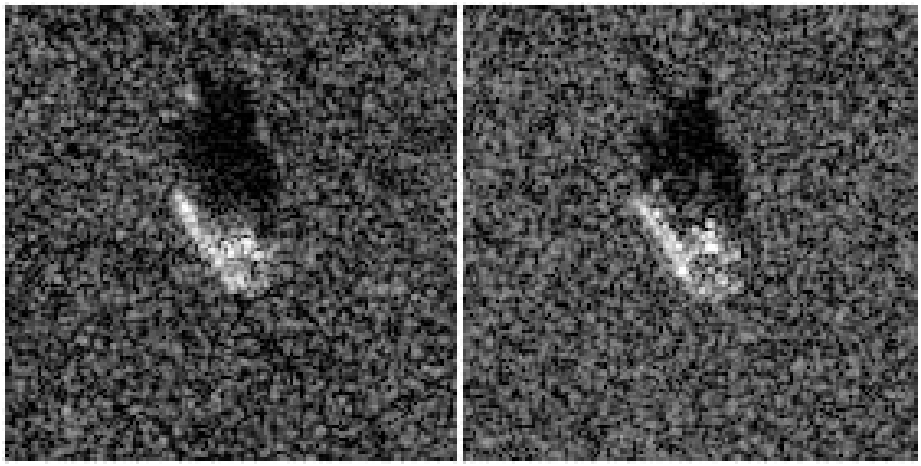


Figure 4.1: Left: MSTAR image of T72 tank. Right: SARSIM simulation of T72 tank. CAD models of exact MSTAR vehicles are not available so the simulated image is an estimation based on an online available model of same type.

generic object representations on our simulated dataset that can be transferred to a real SAR image problem. We show that the simulated objects do not have to be replicas of the objects in the objective dataset on a vehicle classification task. The CAD models used for simulations in our experiments are high quality available models from the different CAD model sharing web pages. Our classification accuracy on MSTAR improves, especially under conditions with little training data available as seen on Figure 4.2. Further, we show that the training time can be reduced when a network is pre-trained on simulated data.

As MSTAR does not contain many of the real world challenges of operational SAR ATR, the experiments have potential for further development. With the literature on transferability of ImageNet in mind, it suggests that different classification and interpretation tasks on SAR data could benefit from pooling simulated datasets across a range of object categories. Land cover classification often looks for features in SAR images that distinguish city areas from e.g. mountainous or agricultural areas, but these features might as well be shared with algorithms for classifying vehicles. The potential for SAR ATR models trained on a wider range of object categories might also be larger as robustness is accumulated with a model trained within a broader context. The SARSIM data has been made publicly available to encourage future work in SAR object recognition to train on high-quality simulated data [KDS].

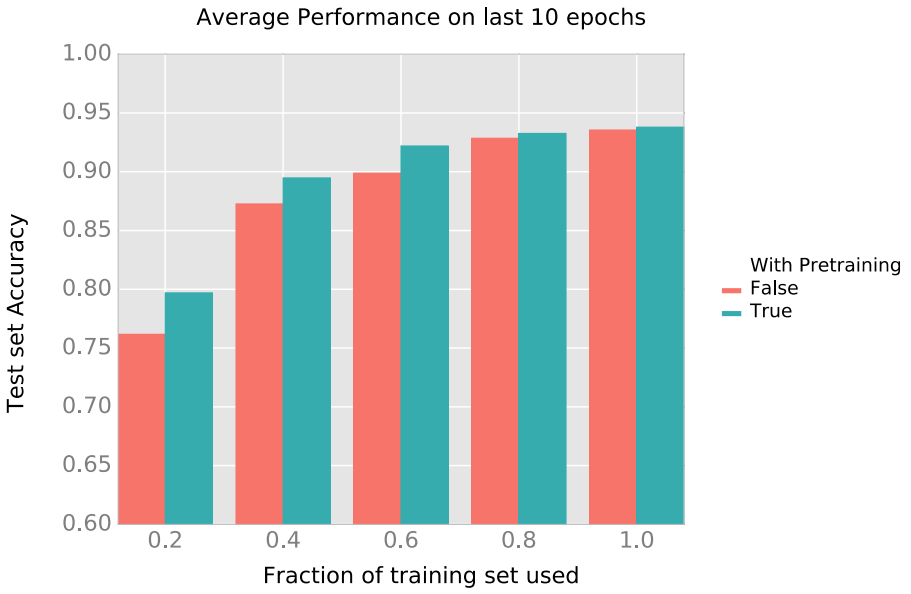


Figure 4.2: Test performance after training on different fractions of the MSTAR dataset, with either pre-training on SARSIM or random initialized network parameters.

4.2 Modeled Target Data

Predictive analysis is a key element in many remote sensing applications, such as meteorology, climatology, geo-science, and several aspects of intelligence. While it is easy to record large quantities of data from satellite borne imaging equipment in Earth observation applications, it is usually labor intensive to obtain ground truth information for it. Often it requires on site measurements in areas that are hard to reach and in best case relatively few point measurements are obtained. Large deterministic models built upon physical information of a system such as weather models, can serve as an approximation of ground truth data in some cases. While not representing the exact truth, these models can supply target values for predictive algorithms on a global scale. This approach is taken in Papers D, E where statistical retrieval of atmospheric parameters is studied. The task is to predict temperatures at 90 different altitudes in the atmosphere from infrared sounder measurements. For this we investigate Convnets as regression models on the infrared sounding images covering the Earth, containing several thousand spectral image bands. Little literature is concerned with Con-

vnets for regression, and in this particular case of multi dimensional input and output regression new challenges are posed. The many spectral bands of infrared sounding data pose a computational and statistical challenge. Dimensionality reduction is therefore an important step in the regression pipeline. In paper D Minimum Noise Fractions (MNF) are studied as an alternative approach to Principal Component Analysis (PCA). The results show that MNF improve the performance of the regression model for any number of decompositions included. Further, the paper shows that a balance between spatial and spectral sampling yields higher accuracy, a conclusion that lead us to explore the spatial feature extraction properties of a Convnet in Paper E. The Convnet approach to atmospheric temperature modeling yielded both 32% RMSE improvement over the commonly used Linear Regression, but also smoother predictions profiles as seen on the transects in Figure 4.3.

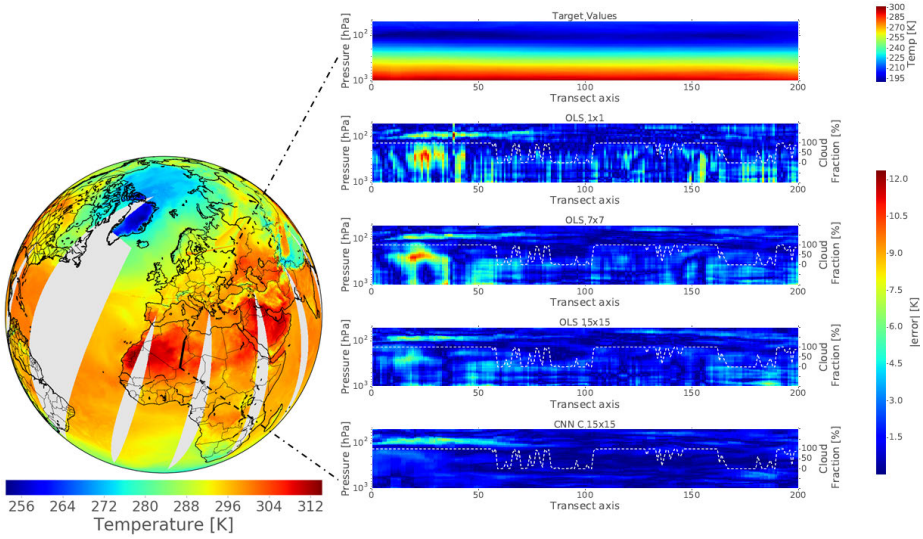


Figure 4.3: Left: Target surface temperature of the test set orbits from the ECMWF weather model. Right: Transect profiles of the atmosphere along a line on Earth, top row shows target temperatures, lower rows show the absolute mean error from 3 different models. Altitudes of the profiles are given in the atmospheric pressure levels. The models tested are Linear regression with single pixel samples, Linear Regression with 7x7 neighborhood pixel samples, Linear Regression with 15x15 neighborhood pixel samples and Convnet with 15x15 neighborhood pixel samples.

A large performance gain was also seen over cloud-marked pixels for the Con-

vnet. This is promising as cloudy pixels often are left out of these predictions due to poorer accuracy. Large spatial coverage is important for the performance of weather models. Inclusion of neighborhood pixels together with the spatial feature extraction properties of Convnets seems to improve accuracy on cloud-marked samples and thereby provide larger spatial coverage.

4.3 Conclusion

In Machine Learning we work within an optimization framework to relate a predicted target variable $\mathbf{y}$, of the target $\mathbf{t}$, from the input variable $\mathbf{x}$ with some parameterized function $\mathbf{y} = \mathbf{f}(\mathbf{x}; \mathbf{W})$ with parameters $\mathbf{W}$. This chapter provides insights into exchanging $\mathbf{x}$ or $\mathbf{t}$ with simulations to provide additional data in situations where it would not be possible to obtain it from direct measurements. Paper C concerns the simulations of input data, $\mathbf{x}$, since little real SAR ATR data is publicly available. As accurate CAD models are rarely available for the exact vehicles in a SAR ATR problem, the concept of Transfer Learning makes use of available vehicle models from online 3D CAD communities. The findings in Paper C are very practical and offer a useful framework for SAR ATR. To further close the gap between operational SAR ATR data and the simulations, it is likely that SARSIM could be extended with more simulations. Extra simulated data is needed both in terms of additional vehicle models to create higher intra class variance, but also with more complicated background scenarios, such as including trees or other vehicles in the background. Experiments on the influence of simulation quality on performance is relevant future work, however, the limited MSTAR is not optimal for these tests. Datasets with generally broader context and variance in e.g. background would serve as a better foundation of evaluating simulation quality.

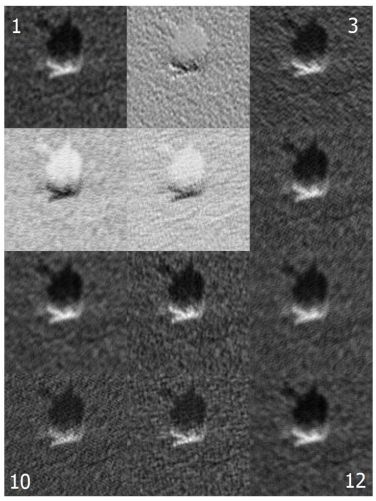
Paper E works with simulated targets, $\mathbf{t}$, since collecting enough real measurements to match the radiance images from the IASI instrument is infeasible. The model we train will be an emulator of the model that provides the target variables. Therefore, we cannot expect to capture physical phenomena not included in the model which provides the target. Despite this, our emulator model can still be useful for supplying global coverage of atmospheric temperature profiles. Since increased coverage in itself reduces uncertainty in numerical weather models, the result is useful. We also show in Paper E how the error in cloud covered areas can be largely reduced by using a Convnet. In general, simulated data can provide useful foundation for experiments when real data is scarce, but it is important to know the limitations and approach the problem with the most suitable framework.

Chapter 5

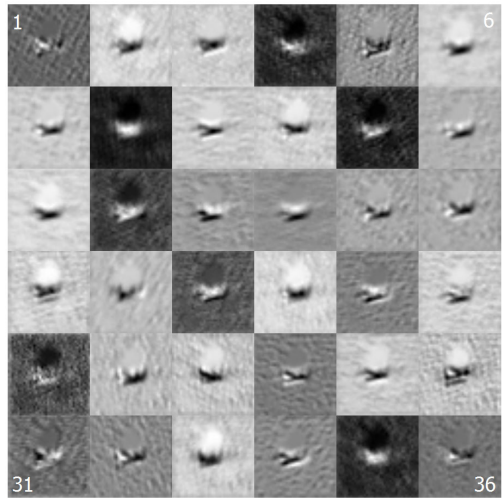
Interpretations

The deep structure of modern Convnets makes it difficult to interpret the individual layer's functionality. Each layer adds to the level of abstraction in the data representation, resulting in increased interpretation difficulty. The high number of layers is therefore both the strength and weakness of Convnets. Being able to interpret and understand is important, as it can reveal properties about data generation (sensors), collection (datasets and their biases) and modeling (being able to build robust models).

The most straightforward way to gain insights about a Convnet is to visualize its internal representations as images. Convolutional layers work as image filters, and thereby preserve the spatial structure in the output. This fact makes it possible to show its transformation of the input as an image as opposed to fully connected layers' outputs. Most classification Convnets contain pooling layers as well, which reduces the resolution of these images, hence the approach works best for early layers. Examples of MSTAR image projection by the first and second layer of a Convnet trained for MSTAR classification can be seen in Figure 5.1. While it seems like the first Convolutional layer mostly perform noise reducing or smoothing on the images, the second layer's filters strongly enhance certain features. Some enhance the shadow and some the target, while another enhances the boundaries between shadow and target or target and background. The SVD+SVM classification approach presented in Section 3.1 also performs filter projections of the input image before the Support Vector Machine classification.



(a) Output from the first Convolutional layer.



(b) Output from the second Convolutional layer.

Figure 5.1: MSTAR SAR image of a T72 tank projected through first two convolutional layers of a trained Convnet. The full Convnet architecture is described in Table 3.1.

Examples of these projections are shown in Figure 5.2. While the filters do not exactly match the Convnet projections, there are similarities, e.g. projection 2 in Figure 5.1b and projection 1 in Figure 5.2. The similarities suggest that some of the features a Convnet uses to classify the MSTAR vehicles are represented by maximum variation of local pixel neighborhoods, i.e. the Eigen filters found by the SVD decomposition of local patches.

While this type of visualization shown here is useful in some cases it is insufficient when moving onto deeper layers of the networks. In deeper layers the abstraction of filters makes it complex to interpret. Another challenge when visualizing deep layers is that the number of filters usually grow. Instead of showing 12 outputs as in Figure 5.1a we might have hundreds or even thousands for some modern Convnet architectures, e.g. as the network in [SVZ13].

5.1 Occlusion Maps

In the concrete case of SAR ATR on MSTAR, interpretations are of great importance to gain new knowledge. Since we know the dataset is not a subset

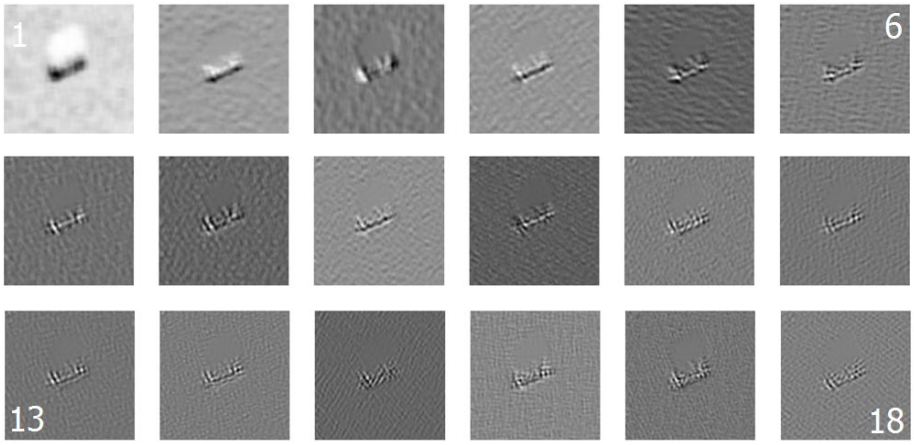


Figure 5.2: SVD filter projections from the SVD + SVM classification approach explained in Section 3.1 of an MSTAR SAR image showing the BTR60 vehicle.

of how objects appear in an operational scenario, we cannot consider training a Convnet on MSTAR solving SAR ATR. On the other hand, learning how an MSTAR trained Convnet represents the data can explain whether it would generalize to an operational scenario. One well known limitation of MSTAR is the stationary background during data collection, i.e. background correlation between training and test data. Each object has remained on its exact location during collection and different azimuthal viewing angles have been obtained by flying by from different sides. This fact is worrying since possible patterns from stones, wheel patterns in the surface, or similar, will be correlated with a specific label class between test and training set. In [SR04] the background was shown to influence the classification accuracy in a positive manner, indicating the problem of stationary backgrounds. The risk is our model learns to recognize these patterns rather than features on the objects and thereby will not generalize to operational data. In order to find out whether this happens with a trained Convnet we can visualize parts of the images that are assigned high relevance. Visualizing predictions and internal representations from Convnets has recently gained increased interest. Several approaches to, e.g. determining which pixels in the input image are of highest relevance to the prediction have been proposed [ZF14], [BBM$^+$15], [SVZ13], [SDV$^+$16]. One approach is to produce occlusion maps as proposed in [ZF14]. In this method we find an image that our model correctly classifies with high confidence. By iteratively occluding the image with a small mask on top of every pixel position and evaluating the probability of the correct class, we obtain a map that directly shows the occlusion's effect on the model's confidence. Further, we can plot a discrete label map of the classes

which the model predicts during occlusion. This reveals the positions where the model's confidence is lowered to such a degree that a misclassification happens. Examples of MSTAR image occlusion maps can be seen on Figure 5.3, while a larger set of occlusion masks can be found in Appendix B. The method from [ZF14] has been adjusted to fit the SAR image domain. Rather than a gray occlusion mask, we use a patch of background pixels as occlusion mask. While a gray mask might be neutral in the natural image domain, it is not in SAR images due to their pixel value distribution. Further, we use a 10 pixel diameter circular occlusion mask to create smooth probability maps.

The maps on Figure 5.3 show different learnings. First of all they all seem invariant to occlusions in the background. This indicates that our model suffers less from overfitting to background features than the study in [SR04]. On Figure 5.3a we see a circular pattern in the probability map with twice the diameter of our mask. This pattern arises when one, or few, pixels are very important for the model confidence. It is generally worrying that one pixel is important for model generalization, however there is a natural explanation in the case of MSTAR. MSTAR is an identification task of 10 specific vehicles, and it is therefore not strange if significant scatters of some targets make them easy distinguishable. In classification tasks where a certain object variance within classes is present, the model behavior might be different. If e.g. we are classifying vehicles into general classes like "tanks", we would expect the cannon to be a significant feature, but in MSTAR there are four tanks that we must learn to distinguish. The ZSU23-4 vehicle on 5.3a has a parabola disk mounted on its top which yields a scatter significantly different from all other targets (see picture in Appendix A.1b). The ZIL131 truck in the occlusion map on Figure 5.3c is the tallest target in the MSTAR dataset. Since the shadow cast in SAR images is dependent on the target height we find a clear drop in confidence when the shadow is decreased in the upper part. In occlusion maps the model confidence serves as quantification of the importance of image regions. One can also count number of occluded pixels that leads to misclassification and by this get a measure of model occlusion sensitivity across datasets. This is shown in Tables 5.1 and 5.2 for target and shadow pixels individually. The pixel annotation masks developed in Paper B are used to mark which pixels belong to the object and which to the shadow.

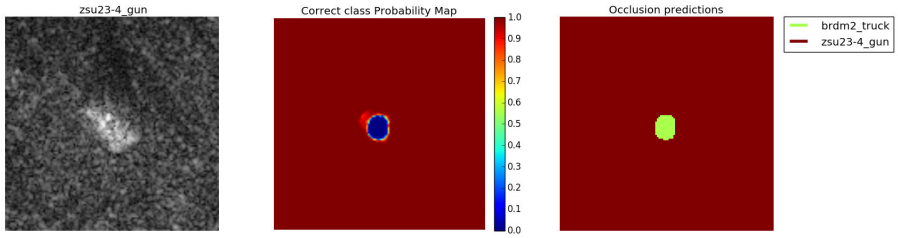
The elements of Table 5.1 and 5.2 represent the percentage of pixels that falls into each class when occluded. Low values in the diagonals are results of the model being sensitive to occlusions in the given area. We can conclude that the model is generally more sensitive to occlusions of the target rather than the shadow.

Table 5.1: The occlusion confusion matrix presents the classifications when occluding pixels marked as *targets* in the annotation masks. The diagonal elements are the percentage of times where occluding target pixels did not have a negative effect on the models classification. Low diagonal values indicate therefore that our model is sensitive to target occlusion.

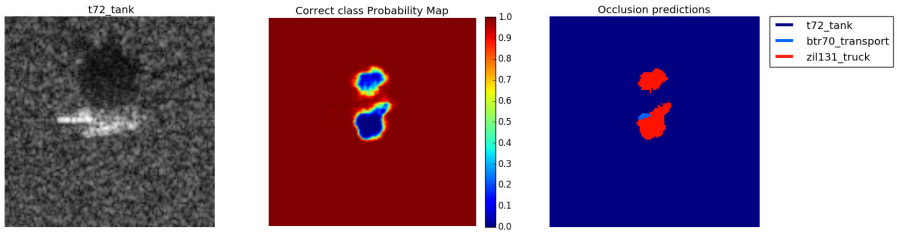
	t72 tank	bmp2 tank	btr70 transport	btr60 transport	2s1 gun	brdm2 truck	d7 bulldozer	t62 tank	zil131 truck	zsu23-4 gun
t72 tank	45	17	51	01	00	03	00	00	05	06
bmp2 tank	00	71	32	02	00	00	00	00	00	00
btr70 transport	00	03	96	02	00	00	00	00	00	00
btr60 transport	00	03	25	70	01	00	00	00	00	00
2s1 gun	00	04	20	01	69	16	00	00	02	00
brdm2 truck	00	00	00	00	00	100	00	00	00	00
d7 bulldozer	00	00	00	00	00	25	57	00	08	06
t62 tank	02	00	02	00	00	95	00	21	04	21
zil131 truck	00	00	00	00	00	02	00	00	98	00
zsu23-4 gun	00	00	00	00	00	21	00	00	04	81

Table 5.2: The occlusion confusion matrix presents the classifications when occluding pixels marked as *shadow* in the annotation masks. The diagonal elements are the percentage of times where occluding shadow pixels did not have a negative effect on the models classification. That the diagonal elements are generally high means that the model is not very sensitive to occlusion of the shadow for most of the classes.

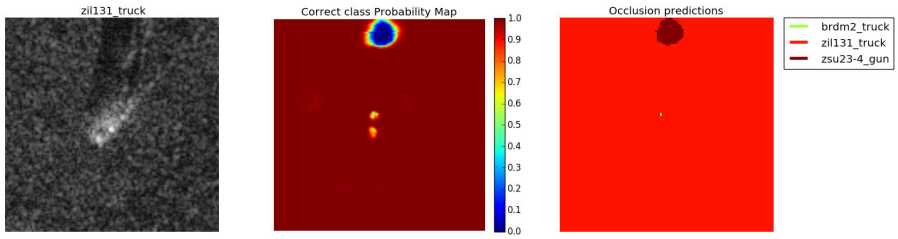
	t72 tank	bmp2 tank	btr70 transport	btr60 transport	2s1 gun	brdm2 truck	d7 bulldozer	t62 tank	zil131 truck	zsu23-4 gun
t72 tank	78	01	02	00	01	00	00	00	09	08
bmp2 tank	00	91	03	00	00	07	00	00	00	00
btr70 transport	00	01	98	00	00	00	00	00	01	00
btr60 transport	00	01	03	93	01	00	00	00	02	00
2s1 gun	00	00	01	00	93	01	00	00	04	00
brdm2 truck	00	00	00	00	00	100	00	00	00	00
d7 bulldozer	00	00	00	00	00	06	82	00	02	10
t62 tank	00	00	00	00	00	10	00	57	04	30
zil131 truck	00	00	00	00	00	00	00	00	98	03
zsu23-4 gun	00	00	00	00	00	01	00	00	00	99



(a) ZSU23-4 Anti Aircraft Vehicle



(b) T72 Tank.



(c) ZIL131 Truck.

Figure 5.3: Left: original MSTAR image. Center: Probability of correct class for each position of the occlusion mask. Right: color map where each color represents the class that the network assigns the highest probability, for each position of the occlusion mask.

5.2 Layer Activation Clustering

The idea of stacking several layers in Convnets is to break up feature representations in levels of increasing complexity. With this hierarchical structure early

layers consist of general representations that can be used to describe image information across a broad range of context and deep layers become more specific to certain classes. This type of image context representation allows for reuse of features but whether this happens with trained networks is hard to prove. [ZF14] proposed a method named *deconvolutional networks* as an approach to visualizing feature extractions from layers in a Convnet. In their method an image is first forwarded through a Convnet, then a layer node is selected for visualization and the node's output is back-propagated into input image space by inverted operations. This produces visualizations of the information left in a deep layer with same spatial resolution as the input image. The Convnet architecture used in [ZF14] has 1376 feature maps and additional 4096 neurons in the fully connected layers. This high dimensionality poses a challenge regarding selecting relevant maps to visualize and this is the limitations of most visualization techniques. In Paper F we propose an unsupervised method to cluster all feature representations in each layer of a network and produce a discrete label map from the result. This leaves the user with much fewer visualizations to interpret. The contributions of Paper F are twofold. First, we propose a pipeline for feature map clustering that compresses the amount feature map visualizations from possibly thousands to one per layer. Second, we use the proposed method to explain why Transfer Learning with Convnets works very well despite diversity of the datasets on which the model transfer is performed.

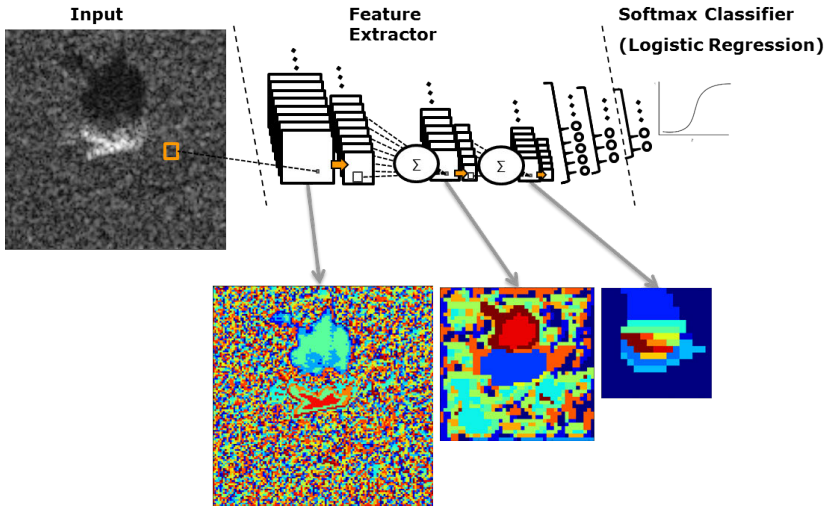


Figure 5.4: The figure shows the labelmaps for an MSTAR image, for each of three different convolutional layers in a Convnet trained on MSTAR.

The proposed scheme to compress feature map visualizations considers each node in a layer an element of an activation vector. A node is either a feature map from a convolutional layer or a neuron from a fully connected layer. Each layer produces activation vectors from all nodes in the given layer, which can be clustered to find groups in the activations. We use the Dirichlet Process Gaussian Mixture Model (DPGMM), [BJ+06], clustering scheme as it has two important properties for this application. First, it assumes underlying mixtures to be Gaussian distributed clusters which suit this problem well assuming there is some variance in the image representations. Secondly, it initializes an infinite amount of mixtures (in practice a maximum number must be selected) with priors sampled from the a Dirichlet process. These priors will have rapidly decreasing probability and during optimization of the data likelihood many of them will not be assigned any points. This provides an unsupervised clustering scheme that yields as few clusters as possible to fit the data. We find this property well suited for our application since the number of feature representations inside the networks are unknown. While feature maps consist of many activation pixels per image, the neuron from a fully connected layer yield one vector per image. Two different approaches to visualization and interpretation are therefore taken. The result of clustering feature map activation pixels can be restructured into label color maps for selected images as shown on Figure 5.4. The output of a fully connected layer has per definition pooled all spatial information out of its representation of input data. Instead of interpreting specific feature representations, we suggest to interpret fully connected layers by their ability to cluster a set of images and analyze the context that is preserved in the resulting clusters. The results in Paper F show that meaningful context representations exist in deep layers of an ImageNet trained networks when analyzing microscopic cell images that the network never saw during training. It explains why Transfer Learning works well for even distinctly different datasets, and shows that it is a large part of the network that contributes to the performance gain. Figure 5.5 summarizes the results from Paper F as label maps generated from a stained gland cell samples shows meaningful interpretation inside a Convnet that was trained on ImageNet data, which significantly differs from the cell image. Some meaningful context representation on the cell images is still present in very deep fully connected layers. The result of clustering all cell images by the vector representation from a fully connected layers can be seen on Figure 5.6. Cluster 1 is mainly benign samples and cluster 2 mainly malign.

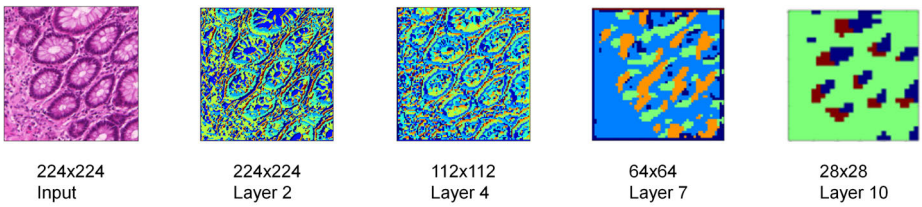


Figure 5.5: Label maps for different convolutional layers in a Convnet trained on ImageNet. The image passed through the network is a stained gland cell sample which is significantly different data than what the network was trained on. Data is from the Warwick QU datasets described in [SSR15, SPC+17].

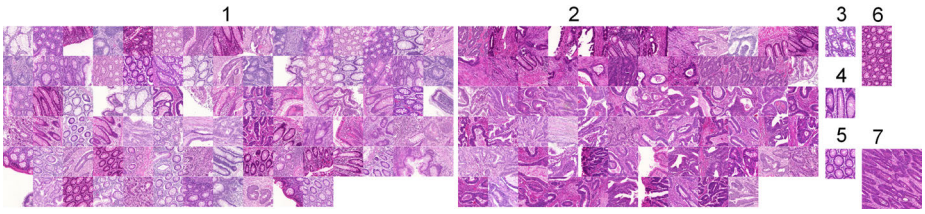


Figure 5.6: Clustering result on image vectors from layer 14 (fully connected) out of 16 layers. The test images are clustered with some context preserved so the image representation in this layer is likely to be transferable.

5.3 Conclusion

There are two challenges when interpreting Convnets with visualization techniques. The first is to quantify model sensitivity over a set of images, as opposed to only visualizing single image representation. Secondly, the high dimensionality of feature space inside a network leads to the practical issue of manually verifying the nodes of the network. In this chapter both challenges are addressed. Occlusion maps offer a quantification of a model's sensitivity measured by its confidence. The technique can be adapted to fit other image modalities by changing occlusion masks to statistically neutral values according to the distribution. Further, if segmentation masks are provided we can quantify a model's occlusion sensitivity with respect to individual context categories. High feature dimensionality in Convnets can be reduced by considering clusters in feature space. Clustering activations from layers in a Convnet result in a reduced visualization space and can be used to interpret the context by means of the clusters' representation of input data. The proposed activation cluster-

ing scheme is a general framework and in our case it proved useful explaining Transfer Learning between dissimilar datasets.

Chapter 6

Discussion and Conclusion

Convnets have a strong ability to represent image context by stacked feature representations proven by the vast amount of recent literature, experiments and their use. However, regarding problems related to limited datasets and interpretations of the models, there are still issues that need to be addressed further. In this thesis a set of tools have been developed for coping with dataset limitations. The tools are based on changing experimental setups, using simulated input and output data when real data is sparse and interpreting model functionality by visualizations. Experimentally, the contributions of this thesis concern obtaining information when dealing with limited datasets. Promising solutions were found when using simulated data for SAR ATR, a field where data collection is limited by high costs. These results encourage future research to look in this direction.

Within the domain of SAR ATR this thesis contributed with several methods to tackle limited datasets. Paper A investigated translational invariance for classification models to show the algorithm's generalizability to an operational scenario with higher translational variance. While translational invariance can be an important parameter, the experiments need to be extended in order to conclude on optimal classifiers for SAR ATR. The largest problem is that the SAR ATR data foundation generally is weak. Paper C showed good results by using Transfer Learning between simulated data and the MSTAR dataset. This proves that simulated data can play a larger role in future SAR ATR and possi-

bly strengthen the data foundation. To account for the limitations in MSTAR such as lack of depression angles and intra-class variation the SARSIM simulated dataset was extended on these parameters. However, there were several other parameters on which the dataset could be improved that would further reduce the gap to operational SAR ATR data. Complex backgrounds with trees and other vehicles would introduce more variance and challenge classifiers in a realistic way. Another general limitation within SAR ATR is the separation of classification/recognition from detection of objects. If we consider the large areas of ground a SAR sensor can cover, new studies on finding areas of interest, segmenting land cover types and detecting objects to pass to a classifier would also be relevant. In Computer Vision the tasks of segmenting or detecting objects have merged with classifying objects, e.g. in Semantic Segmentation, and Convnets fully support this concept. In SAR ATR, little data exist for testing algorithms that e.g. both detect and classify objects in large maps, but this could be a focus in future work and data collection campaigns.

Dense predictions of targets were performed in both Paper B and Paper E on two different tasks with good results. Paper **??** concerned predictions over all pixels in a SAR image with categories target, shadow and background. Paper E the a pixel-wise predictions of atmospheric temperature profiles over infrared spectral measurements. However, the approach of reformulating a Convnet to work on patches in a pixel-wise manner is computationally expensive. For all patches that partly overlaps, the Convnet performs some or more redundant convolutions due to the overlap of the patch windows. With smaller patch sizes the redundancy decreases hence whether the computational load is a problem depends on the specific application and model design. It is not in the scope of Paper B to include experiments on optimal patch size. For the small MSTAR images (128x128 pixels) a patch size of 33x33 pixels is not a problem since an image can be processed in a few seconds, but for larger SAR maps this might become a problem. Newer architectures for dense predictions with Convnets trade the density of convolutions in a patch for a Convnet architecture that produces larger predictions maps in one go with significantly less computations, [YK15]. The receptive field in the approach of [YK15] from where information is aggregated into a convolutional layer output can be designed to be the same as in the patch based approach. Further, spatial resolution is kept so sharp edges in segmentation masks are well modeled as opposed to other dense prediction methods, e.g. [LSD15]. The Convnet architecture in [YK15] could be used for both the segmentation task in Paper B and regression problem in Paper E if the network output layer is adapted to the respective task.

The experiments in Paper E combined with the dimensionality reduction techniques in Paper D show great potential for Convnets in retrieval of atmospheric parameters. To further extend on the work in Paper E two main concepts would be interesting to explore. First, as an alternative to the Minimum Noise Frac-

tion algorithm in Paper D, the first layer in the Convnet architecture could as well perform dimensionality reduction. By convolving a kernel across the spectral channels in the Infrared Sounder data, dimensionality reduction could be learned simultaneously with the regression task. The second extension would be to include the spatial relationship between the target temperatures as they are expected to be highly correlated with neighbors in both vertical and horizontal direction. This could be done by exchanging the least square regression error function that assumes target independence. Possible alternative error functions that include information from neighboring targets could be based on Markov Chains or Markov Random Fields.

Visualization of a Convnet's image representation offer intuition about its functionality. This is an important tool when evaluating generalizability of a model. Occlusion maps are one way of gaining insight into the generalizability as it provides us with a direct pixel-wise measure of model confidence when a local area is occluded. The occlusion map experiments on MSTAR revealed that our model had learned to recognize a vehicle by a single strong reflection on it. This fact is worrying for generalizability if we suspect a similar reflection could be present on other vehicles given a larger set of vehicle types.

When it comes to understand the internal data representation in Convnets the main challenges are the abstraction and dimensionality of the representation. The label activation map presented in Paper F is a direct way of reducing dimensionality when considering visualizations of Convnets. Further, the label activation maps consider clusters of Gaussian mixtures on a Convnet's image representations a way of measuring its relevance to the given data. This was found very useful for explaining Transfer Learning between dissimilar datasets.

From the results presented in this thesis we find several tools and approaches that can explore a model's generalizability when working with limited datasets. Generally one can test whether certain problems exist by visualization and use Transfer Learning and simulated data to tackle these. Alternatively, it might be beneficial to scope the experimental setup to cope with expected variances not covered by the dataset. All these findings provide a foundation for building practical solutions with Convnets for Computer Vision problems.

Bibliography

[AWZ02] Elmehdi Aitnouri, Shengrui Wang, and Djemel Ziou. Segmentation of small vehicle targets in SAR images. In *AeroSense 2002*, pages 35–45. International Society for Optics and Photonics, 2002.

[BBM+15] Sebastian Bach, Alexander Binder, Grégoire Montavon, Frederick Klauschen, Klaus-Robert Müller, and Wojciech Samek. On pixel-wise explanations for non-linear classifier decisions by layer-wise relevance propagation. *PloS one*, 10(7):e0130140, 2015.

[BFG+99] Monique Bernier, Jean-Pierre Fortin, Yves Gauthier, Raymond Gauthier, René Roy, and Pierre Vincent. Determination of snow water equivalent using radarsat sar data in eastern canada. *Hydrological Processes*, 13(18):3041–3051, 1999.

[Bis06] C.M. Bishop. *Pattern Recognition and Machine Learning*. Princeton Legacy Library. Springer, 8th edition, 2006.

[BJ+06] David M. Blei, Michael I. Jordan, et al. Variational inference for Dirichlet process mixtures. *Bayesian analysis*, 1(1):121–144, 2006.

[C+15] François Chollet et al. Keras. `https://github.com/fchollet/keras`, 2015.

[CSD+98] Erik Lintz Christensen, Niels Skou, Jørgen Dall, Kim Wildt Woelders, Jan Hjelm Jorgensen, Johan Granholm, and Soren Norvang Madsen. Emisar: An absolutely calibrated polarimetric L-and C-band sar. *IEEE Transactions on Geoscience and Remote Sensing*, 36(6):1852–1865, 1998.

[DS83] TS Durrani and KC Sharman. Eigenfilter approaches to adaptive array processing. In *IEE Proceedings F (Communications, Radar and Signal Processing)*, volume 130, pages 22–28. IET, 1983.

[GB10] Xavier Glorot and Yoshua Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[HSS12] Geoffrey Hinton, N Srivastava, and Kevin Swersky. Lecture 6a overview of mini–batch gradient descent. *Coursera Lecture slides https://class.coursera.org/neuralnets-2012-001/lecture,[Online*, 2012.

[HW62] David H Hubel and Torsten N Wiesel. Receptive fields, binocular interaction and functional architecture in the cat's visual cortex. *The Journal of physiology*, 160(1):106–154, 1962.

[HWH16] Shiqi Huang and Ting Zhang Wenzhun Huang. A new SAR image segmentation algorithm for the detection of target and shadow regions. *Scientific reports*, 6, 2016.

[IS15] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In *International Conference on Machine Learning*, pages 448–456, 2015.

[KAD16] Anders Kusk, Adili Abulaitijiang, and Jørgen Dall. Synthetic SAR image generation using sensor, terrain and target models. In *EUSAR 2016: 11th European Conference on Synthetic Aperture Radar, Proceedings of*, pages 1–5. VDE, 2016.

[KB14] Diederik Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[KDS] Anders Kusk, Jœ rgen Dall, and Henning Skriver. SARSIM simulated SAR dataset. `https://zenodo.org/record/573750#.WXHMPGcUnq5`. Published: May 10, 2017.

[KSH12] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.

[LBD+89] Yann LeCun, Bernhard Boser, John S Denker, Donnie Henderson, Richard E Howard, Wayne Hubbard, and Lawrence D Jackel. Backpropagation applied to handwritten zip code recognition. *Neural computation*, 1(4):541–551, 1989.

[LHB04] Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–104. IEEE, 2004.

[Low99] David G Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

[LSD15] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.

[MCSD91] Søren Nørvang Madsen, Erik Lintz Christensen, Niels Skou, and Jørgen Dall. The danish sar system: Design and initial tests. *IEEE Transactions on Geoscience and Remote Sensing*, 29(3):417–426, 1991.

[MHNJ17] David Malmgren-Hansen and Morten Nobel-Jørgensen. Sarbake overlays for the mstar dataset. `https://data.mendeley.com/datasets/jxhsg8tj7g/2`, 2017.

[Mor15] David AE Morgan. Deep convolutional neural networks for ATR from SAR imagery. *Proceedings of the Algorithms for Synthetic Aperture Radar Imagery XXII, Baltimore, MD, USA*, 23:94750F, 2015.

[MP43] Warren S McCulloch and Walter Pitts. A logical calculus of the ideas immanent in nervous activity. *The bulletin of mathematical biophysics*, 5(4):115–133, 1943.

[NGM15] Rickard Norlander, Josef Grahn, and Atsuto Maki. *Wooden Knot Detection Using ConvNet Transfer Learning*, pages 263–274. Springer International Publishing, 2015.

[NH10] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.

[ØKCL16] Nina Ødegaard, Atle Onar Knapskog, Christian Cochin, and Jean-Christophe Louvigne. Classification of ships using real and simulated data in a convolutional neural network. In *Radar Conference (RadarConf), 2016 IEEE*, pages 1–6. IEEE, 2016.

[RHP+94] Y Rauste, T Hame, J Pulliainen, K Heiska, and M Hallikainen. Radar-based forest biomass estimation. *International Journal of Remote Sensing*, 15(14):2797–2808, 1994.

[RHW85] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, DTIC Document, 1985.

[Ros58] Frank Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological review*, 65(6):386, 1958.

[RWV+98] Timothy D Ross, Steven W Worrell, Vincent J Velten, John C Mossing, and Michael L Bryant. Standard SAR ATR evaluation experiments using the MSTAR public release data set. In *Aerospace/Defense Sensing and Controls*, pages 566–573. International Society for Optics and Photonics, 1998.

[SDV+16] Ramprasaath R Selvaraju, Abhishek Das, Ramakrishna Vedantam, Michael Cogswell, Devi Parikh, and Dhruv Batra. Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization. *arXiv preprint arXiv:1610.02391*, 2016.

[She16] Jamie Sherrah. Fully convolutional networks for dense semantic labelling of high-resolution aerial imagery. *arXiv preprint arXiv:1606.02585*, 2016.

[SHK+14] Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*, 15(1):1929–1958, 2014.

[Skr12] Henning Skriver. Crop classification by multitemporal c-and l-band single-and dual-polarization and fully polarimetric SAR. *IEEE Transactions on Geoscience and Remote Sensing*, 50(6):2138–2149, 2012.

[SLJ+15] Christian Szegedy, Wei Liu, Yangqing Jia, Pierre Sermanet, Scott Reed, Dragomir Anguelov, Dumitru Erhan, Vincent Vanhoucke, and Andrew Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 1–9, 2015.

[SPC+17] Korsuk Sirinukunwattana, Josien PW Pluim, Hao Chen, Xiaojuan Qi, Pheng-Ann Heng, Yun Bo Guo, Li Yang Wang, Bogdan J Matuszewski, Elia Bruni, Urko Sanchez, et al. Gland segmentation in colon histology images: The glas challenge contest. *Medical image analysis*, 35:489–502, 2017.

[SR04] Rolf Schumacher and Kh Rosenbach. ATR of battlefield targets by SAR classification results using the public MSTAR dataset compared with a dataset by qinetiq uk. In *RTO SET Symposium on Target Identification and Recognition Using RF Systems*, 2004.

[SSR15] Korsuk Sirinukunwattana, David R.J. Snead, and Nasir M. Rajpoot. A stochastic polygons model for glandular structures in colon histology images. *IEEE transactions on medical imaging*, 34(11):2366–2378, 2015.

[SST99] Henning Skriver, Morten T Svendsen, and Anton G Thomsen. Multitemporal c-and l-band polarimetric signatures of crops. *IEEE Transactions on Geoscience and Remote Sensing*, 37(5):2413–2429, 1999.

[SVZ13] Karen Simonyan, Andrea Vedaldi, and Andrew Zisserman. Deep inside convolutional networks: Visualising image classification models and saliency maps. *arXiv preprint arXiv:1312.6034*, 2013.

[UE90] Fawwaz T Ulaby and Charles Elachi. Radar polarimetry for geoscience applications. *Norwood, MA, Artech House, Inc., 1990, 376 p. No individual items are abstracted in this volume.*, 1, 1990.

[VDLN12] Jacob S Vestergaard, Anders L Dahl, Rasmus Larsen, and Allan A Nielsen. Classification of polarimetric SAR data using dictionary learning. In *SPIE Remote Sensing*, pages 85370X–85370X. International Society for Optics and Photonics, 2012.

[WKC+99] Robert A. Weisenseel, W. Clem Karl, David A. Castañon, Gregory J. Power, and Phil Douville. Markov random field segmentation methods for SAR target chips. *Algorithms for Synthtic Aperture Radar Imagery VI*, 3721(1):11–24, 1999.

[YK15] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015.

[ZF14] Matthew D Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European conference on computer vision*, pages 818–833. Springer, 2014.

Appendix A

Datasets

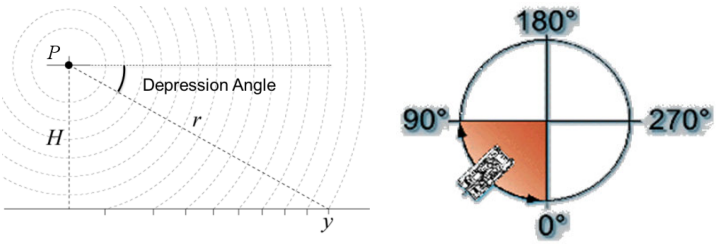
The datasets used in this thesis are referred to as MSTAR, SARSIM, EMISAR and IASI. A short description and sample images are shown in this section.

A.1 MSTAR

MSTAR is a set of real SAR images of military vehicles in 30cm x 30cm resolution. The subset often used for testing classification algorithms contains ten vehicle types recorded at 15° and 17° depression angles. Incidence angles of the radar radio wave on target are characterized by depression angle and azimuth rotation angle illustrated on Figure A.1.

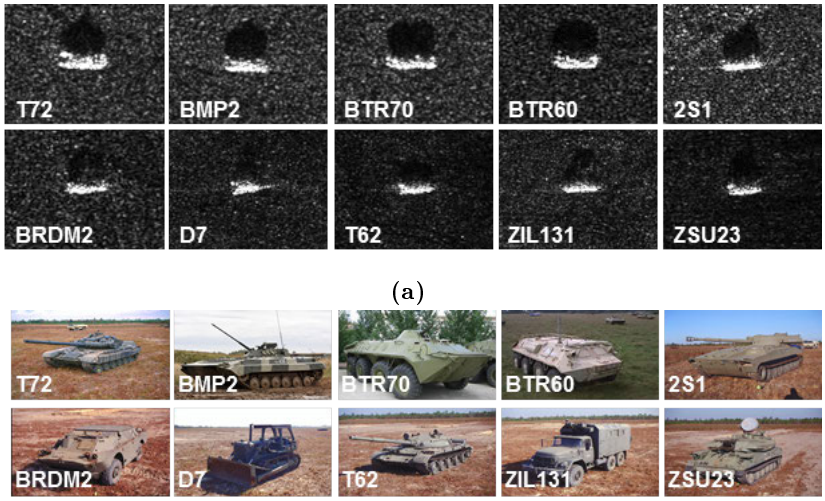
The vehicles are densely sampled in azimuthal rotation with an image for each 1° - 2° depending on the object. The training set contains the ten vehicles at 17° and the test set contains the 15° samples.

Besides the classification set, MSTAR contains additional images of empty backgrounds covering large areas. Additionally, three vehicles in MSTAR have been recorded at depression angles equal to 30°and 45°. These vehicles are a tank - 2S1, an armored patrol car - BRDM2 and an anti-aircraft vehicle - ZSU23-4.



(a) (b)

Figure A.1: Illustration of MSTAR depression angles (a) and azimuthal rotation angles (b).



(a)

(b)

Figure A.2: MSTAR images from the 10-way classification subset. A SAR image of each vehicle is shown (a) at approximately 90°azimuthal rotation. (b) A regular image of each of the objects in MSTAR.

Table A.1: Table of simulation parameters in the SARSIM dataset. In total the dataset contains 21 168 images. "Medium" background is generated with a mean backscatter coefficient of grass and road.

Parameters	Instance	Number
Azimuth Angles	0, 5, ..., 355	72
Depression Angles	15, 17, 25, 30, 35, 40, 45	7
Classes	Tanks, Trucks, Buses, Cars, Humwees, Bulldozers, Motorbikes	7
Models	2 pr. class	14
Background	Grass, Road and Medium	3
Resolution	30cm	1

A.2 SARSIM

SARSIM is a simulated SAR ATR dataset developed by DTU Space in collaboration with Terma A/S. The goal has been to extend on the information about SAR ATR that can be obtained from MSTAR. Rather than ten specific vehicles SARSIM contains seven vehicle classes and each has two instances of vehicles. SARSIM has a 5° azimuthal sampling of objects, but includes seven different depression angles to study the change in object appearance with respect to these. Further, objects have been simulated with different backgrounds. An overview of the settings can be seen in Table A.1.

A.3 EMISAR

EMISAR is a fully polarimetric SAR system developed at DTU Space, i.e. it is capable of transmitting horizontal and vertical polarized radio waves as well as receiving both and do every combination of transmission and receiving these. It has two radar frequencies L-band ($\lambda \in 15 - 30cm$) and C-band ($\lambda \in 3.75 - 7.5cm$). [CSD+98, MCSD91], provides the full description of the technical system details. The dataset referred to as the EMISAR dataset in this paper in a study of vegetation growth with measurements from the EMISAR system. The recordings are of fields near the danish village of Foulum which is the location of the Danish Centre For Food and Agriculture. There are 6 different types of crops marked in the images at Figure A.4, Rye, Grass, Winter Wheat, Spring Barley, Peas and Winter Barley.

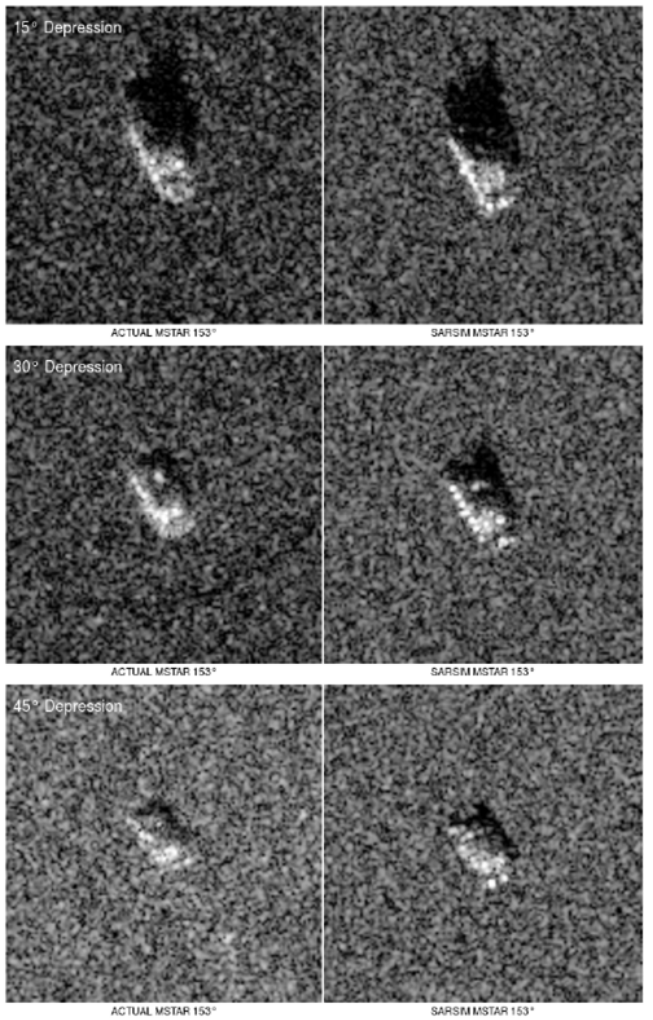
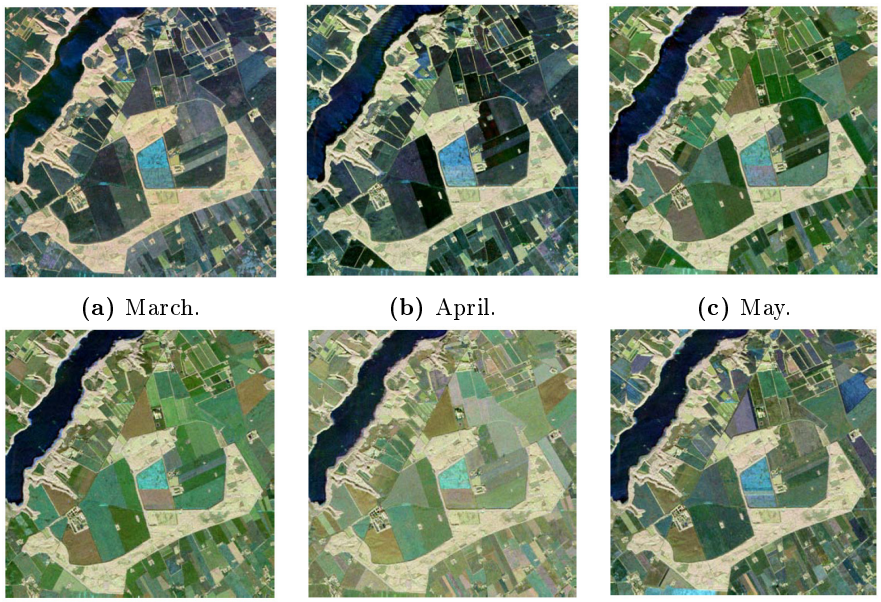


Figure A.3: Left column: Real SAR image from MSTAR of T72 Tank. Middle Column: SARSIM simulation of T90 Tank in 30cm pixel resolution. The rows represent 15°, 30°and 45°depression angles respectively.



(a) March. **(b)** April. **(c)** May.

(d) June. **(e)** July. **(f)** August.

Figure A.4: Color coded polarimetric SAR images from the EMISAR Foulum dataset. EMISAR Foulum dataset is a crop classification dataset with polarimetric SAR images of the whole growth season on certain crop type fields.

A.4 Infrared Atmospheric Sounding Interferometer

The Infrared Atmospheric Sounding Interferometer is an instrument on board the MetOp satellite series. It measures infrared emissions in the spectra from 3.62 - 15.5 μm wavelength (2760 - 645 cm^{-1}), with the intention of providing temperature and water vapor profiles of the atmosphere. The measurements consist of spectra with 8461 bands and more than one million of these are collected each day. Each orbit from IASI produces 1530 scans of 60 points across track and approximately 14 orbits are collected each day corresponding to two global coverages.

For Paper D and E a dataset of 13 orbits from august 2013 were collected and structured in rectangular grids of 1530x60 pixels. These were matched with temperature and water vapor atmospheric profiles from 90 different altitudes.

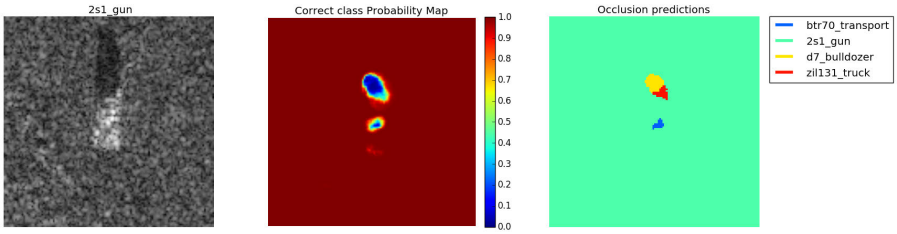
The temperature and vapor profiles are outputs from a weather model by The European Centre for Medium-Range Weather Forecasts (ECMWF). Each orbit has corresponding cloud and land fraction masks associated with it. With these, prediction errors can be analyzed according to e.g. "cloud free" versus "cloudy" predictions.

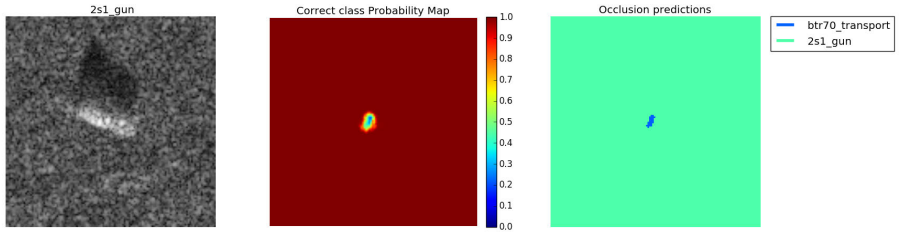
Appendix B

Occlusion Maps

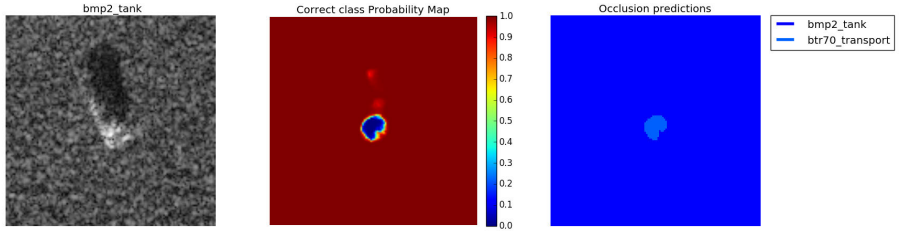
Provided in this appendix are occlusion maps according to the description in Section 5.1 from SAR images in the MSTAR dataset. There are two occlusion maps per vehicle in the MSTAR dataset. The maps support the analysis in Section 5.1, e.g. there seems to be no vehicles where the model is sensitive to features in the background of the image.

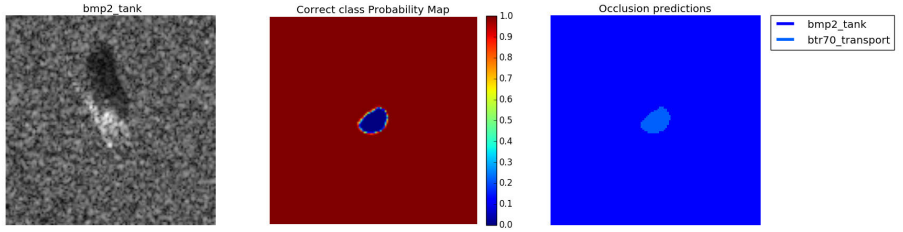
B.1 2s1 Tank



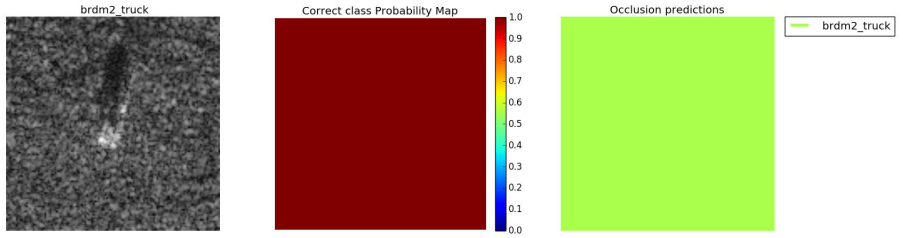


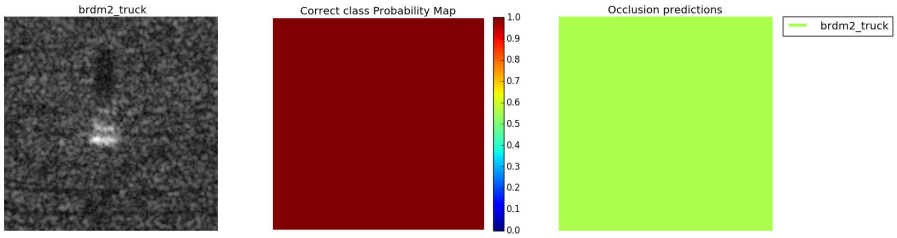
B.2 BMP2 Tank



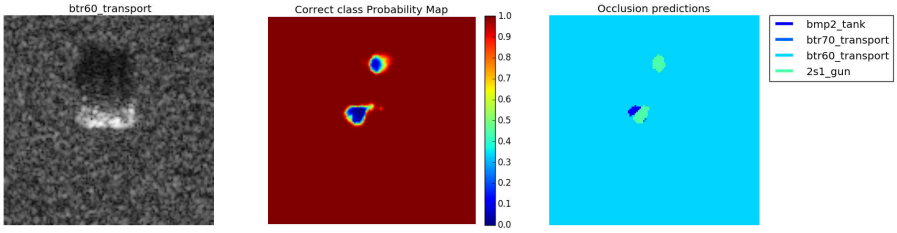


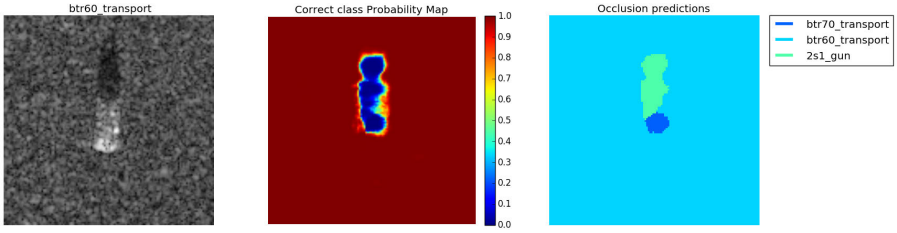
B.3 BRDM2 Patrol Car



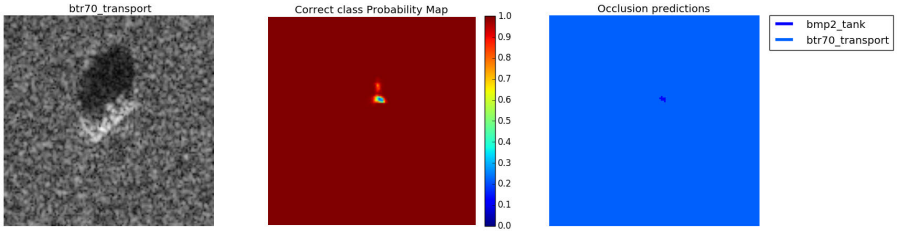


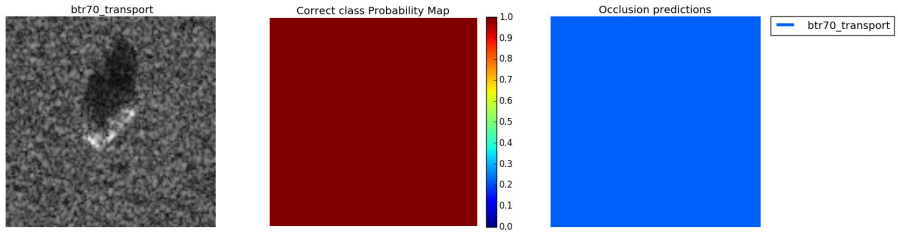
B.4 BTR60 Armored Personnel Carrier



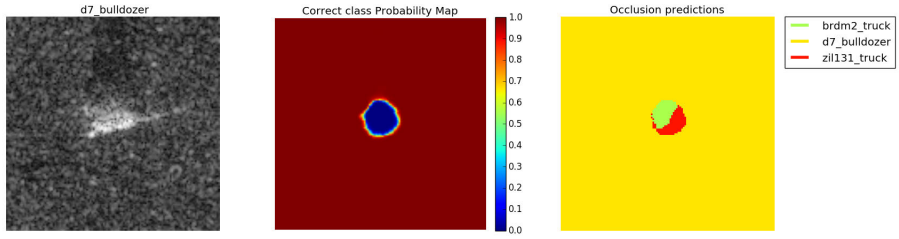


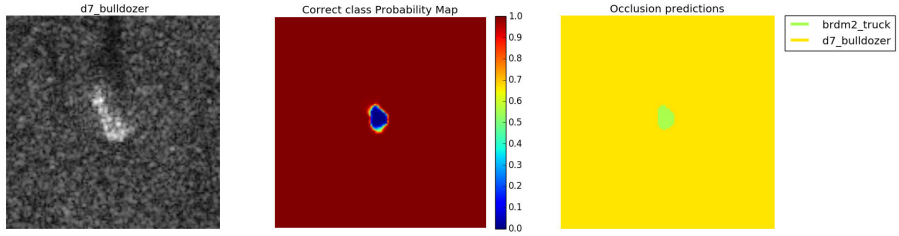
B.5 BTR70 Armored Personnel Carrier



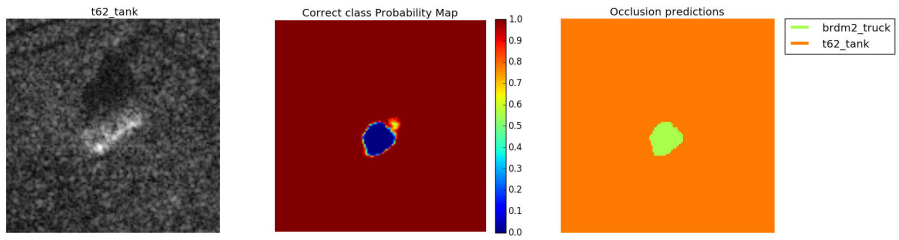


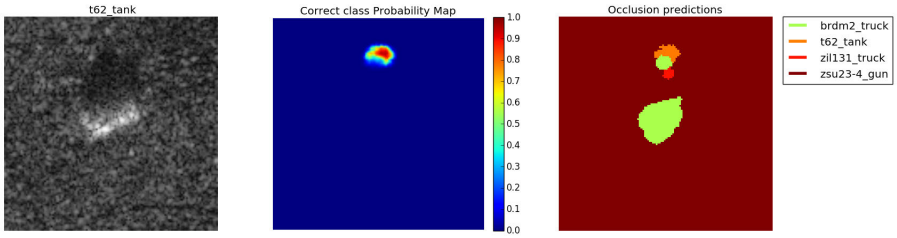
B.6 D7 Bulldozer



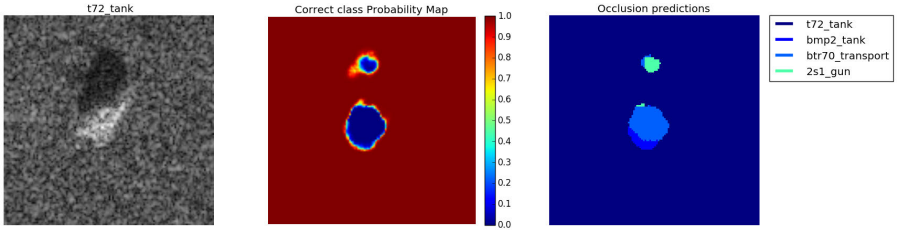


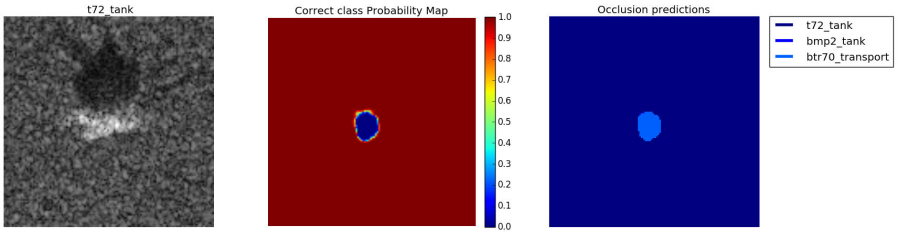
B.7 T62 Tank



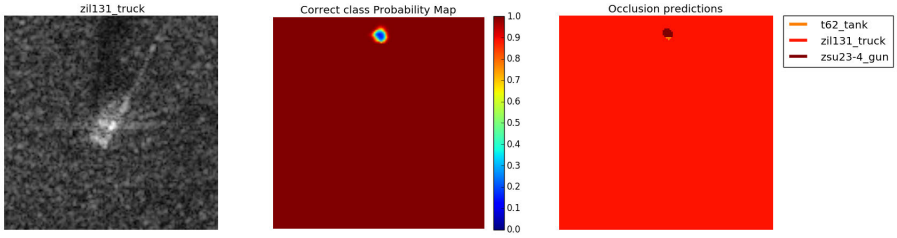


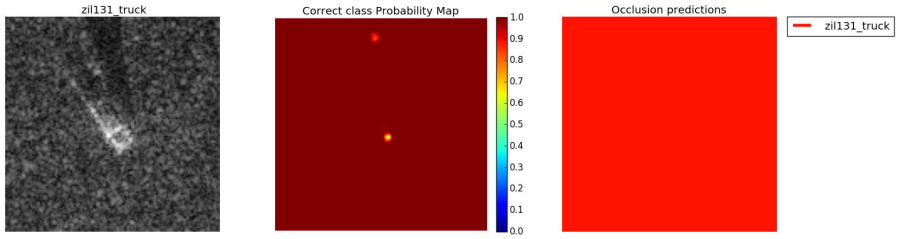
B.8 T72 Tank



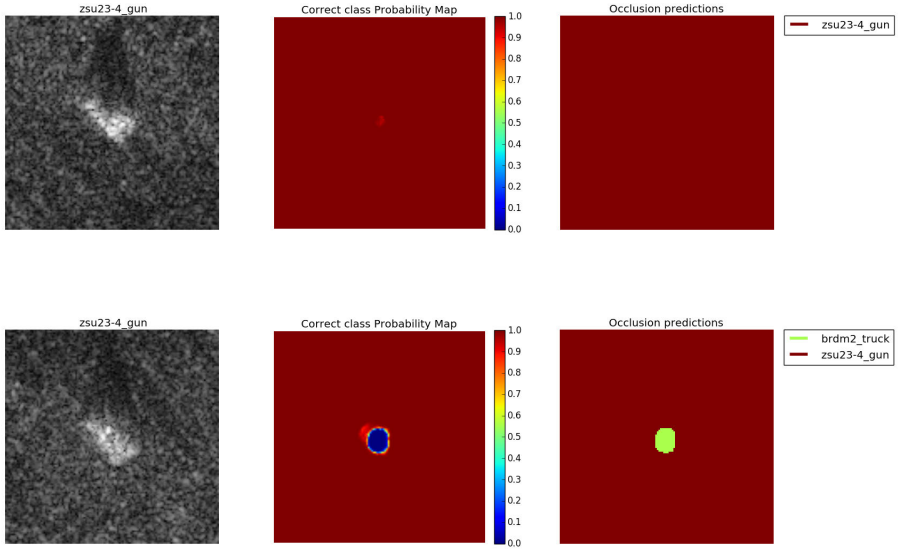


B.9 ZIL131 Truck





B.10 ZSU23-4 Anti Aircraft Vehicle



Appendix C

Publications

Paper A - Training Convolutional Neural Networks for Translational Invariance on SAR ATR

Training Convolutional Neural Networks for Translational Invariance on SAR ATR

David Malmgren-Hansen, Technical University of Denmark, dmal@dtu.dk, Denmark
Rasmus Engholm, Terma A/S, rae@terma.com, Denmark
Morten Østergaard Pedersen, Terma A/S, mdp@terma.com, Denmark

Abstract

In this paper we present a comparison of the robustness of Convolutional Neural Networks (CNN) to other classifiers in the presence of uncertainty of the objects localization in SAR image. We present a framework for simulating simple SAR images, translating the object of interest systematically and testing the classification performance. Our results show that where other classification methods are very sensitive to even small translations, CNN is quite robust to translational variance, making it much more useful in relation to Automatic Target Recognition (ATR) in a real life context.

1 Introduction

Compared to classification studies on natural images, Synthetic Aperture Radar (SAR) datasets are often smaller and contain more biases. One of very few public available SAR Automatic Target Recognition (ATR) datasets is the Moving and Stationary Target Acquisition and Recognition (MSTAR) that was collected during the 1990s. MSTAR contains several subsets with different tasks to solve. One of these is the 10 target classification task where 10 specific vehicles must be recognized. Although it is an interesting problem this does not contain much class variance. A more realistic scenario could be how well general target classes like "tanks" that might contain many different instances can be recognized from other classes of vehicles like "cars" or "busses".

The targets in MSTAR are densely sampled along azimuthal object rotation, but far most images are taken at 15° and 17° depression angles with almost no background variation.

Recent advances in Convolutional Neural Networks (CNNs) for computer vision problems makes it interesting to study the benefits of these algorithms on SAR problems as well. This has been done for the MSTAR dataset by Morgan in [3]. In [3] a CNN classifier is applied to the 10 target classification task from MSTAR and it achieve results similar to other state of the art methods benchmarked on this problem, e.g. by [4], [5], [7] and [8]. Due to the sparsity of object representation in the data, there are more investigations to be made in order to find the best classification models.

In this article we first show how to generate a simulated dataset where the amount of translational variance is bigger than for the MSTAR dataset (see Section 2). We then describe the design of the CNN used as well as the classification algorithms for comparison (see Section 3). Finally we show the results of the comparison (Section 4) and conclude that the convolution layers in a CNN can make it more invariant to translational variance, and superior to other machine learning classification algorithms (Section 5).

2 Data Simulation

For our SAR image simulations, the geometry consists of a CAD model placed on a corrugated surface representing the terrain. The surfaces are generated by selecting points in an ground grid (x-y plane) and then picking the vertical displacement (the z component) from a normal distribution with a given variance. The ground planes vertical displacement is then low-pass filtered to give a smooth and more natural terrain.

A highly simplified optical algorithm is used to trace multiple rays and find the intersection points between the incident signal and the model and/or terrain. The cell value in the image corresponds to the range / cross-range value of the intersection point. Here a simple diffuse scattering contribution is found from the angle between the surface normal at the intersect point and the incident ray direction. Based on whether a ray intersects with the model or terrain, the value of the cell is scaled with a scattering cross section representing the material (metal for the target and soil for terrain).

Parameter values for the simulation model, such as material reflection coefficients and background variation, have been adjusted according to images from the MSTAR dataset. This is done by manually defining boundaries in an MSTAR image for background and target, and then analyse the values from each region. The relationship between the median from the two sets of values was used to define the relation between the reflection coefficients for terrain and vehicle model in the simulation. The terrain topology variation in our simulation is adjusted so histograms from background pixels in a MSTAR image and in a simulated image is most alike.

By adjusting the parameters in this way, we get a simulated image that has a visual appearance close to the real

SAR images, despite using a simple simulation procedure. In **Figure 1** an image from the MSTAR dataset can be seen together with a simulated image from our dataset.

Our simulation tool is implemented in python and GPU accelerated with the VTK toolset. It takes approximately 2 minutes to simulate an image of 192x192 pixel resolution.

(a) MSTAR image of a T62 tank.

(b) Simulated SAR image of a leopard tank.

Figure 1: Comparison of an MSTAR image and a similar vehicle modelled with our SAR simulation algorithm.

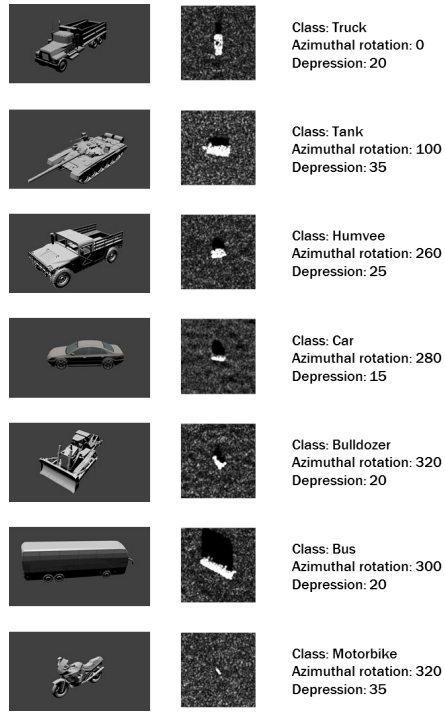


Figure 2: Examples on target instances and their simulated SAR image. Pixel values are clipped in order to show the background in the simulated images.

2.1 Dataset

Our dataset is constructed from 14 CAD models from 7 different vehicle classes, so each class has two instances. When simulating the images, a background is chosen between 100 different generated examples and rotated randomly. Then a model is placed with a given rotation and images from seven different depression angles are generated. A shadow mapping technique is used to remove points not visible from a the radar position so these does not contribute to the simulation. We sample with 20°steps of azimuthal object rotation which gives us 18 images per vehicle per depression angle. This is in total 1764 images of 192x192 pixel resolution and to look most like MSTAR data our pixel spacing is 0.2 meter and resolution 0.3 meter.

CAD model examples from each target class are shown in **Figure 2** along with a simulated SAR image of the given model.

When training our machine learning algorithms we extract sub-images of 128x128 pixels from our dataset. This allow us to gradually extract datasets that have higher translational variance of the target. Our method is illustrated at **Figure 3**. The maximum distance away from the center which our sub image can be extracted is $\frac{192-128}{2} = 32$ pixels in each direction. We extract datasets with different random translation levels within 32 pixels. By gradually increasing the level of random translation we can measure how it effects the accuracy of the algorithms.

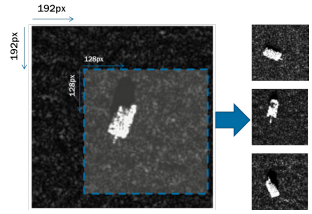


Figure 3: Illustration of how sub-image extraction gives us translational target variance.

3 Classifiers

The classifiers used in our study are all end-to-end learning based models. This means they all take raw pixel values as input and there are no hand crafted feature extractors preprocessing the data for the classifier. One of the nice properties on Convolutional Neural Networks is its ability to learn feature extractors in its convolution layers and base a nonlinear decision boundary on top of this.
The data set described in Section 2.1 is split randomly in five parts on which we make a 5-fold cross-validation of all classifiers. This gives us training set sizes of 1411 samples and test sets of 353 samples for each fold. We report performance of our classifiers as the fraction of correct classifications averaged over the cross-validation. The classifiers has been chosen to span complexity from very simple classifiers (K-Nearest Neigbor) to state of the art (Convolutional Neural Networks and Random Forests). We have futhermore tested various subcomponents of the CNN, the Multi Layer Perceptron i.e. the fully connected layers in the CNN and the Softmax i.e the output layer in the CNN.

3.1 Convolutional Neural Network

Our CNN model has 5 layers with trainable parameters, 3 convolutional and 2 fully connected hidden layers. We train our model with a stochastic gradient descent (SGD) algorithm minimizing the multiclass log-loss over our 7 target classes.

Layer	kernels	kernel size	Dropout fraction
C1	18	9x9	0.25
C2	36	5x5	0.25
C3	120	4x4	0.25
F4	400	120	0.50
F5	7	400	0.25

Table 1: CNN layer sizes shown. In total the model has 138'159 parameters.

After the first two convolution layers we apply a max pooling operator to reduce the dimensionality. The first max pooling operation has a window size of 6x6 pixels and the second a window size of 4x4 pixels. After all layers we apply a Rectified Linear Unit activation function as Krizhevsky et al. in [2] showed how this can make CNN's converge faster. We found it necessary to apply the dropout technique described by Srivastava et al. in [6] to prevent overfitting. In this technique a fraction of randomly chosen network weights are omitted from the weight updates in each iteration of the SGD algorithm. Our model layer sizes together with the fraction of dropout used can be seen in **Table 1**. Considering the amount of variation in target appearance in our dataset, a training set size of 1411 samples cannot be considered a lot and it seems reasonable that measures must be taken to prevent overfitting to the training data.

3.2 Classification models

We use an implementation of Random Forest [1], an ensemble method of classification trees from the Python library scikit-learn. The primary parameters for the Random Forest is the number of trees N and the number of features per tree m. The parameters (N, m) have been found by 5-fold cross-validation to be 512 trees and 128 features per tree.

K Nearest Neighbours is the classic approach of storing all training images together with their class. At test time, euclidean distances is calculated between the given test images and all training images and the k shortest distances vote which class the test images belongs to. the k-parameter have found by 5-fold cross-validation to be optimal at $k = 1$.

A simpler neural network where the input layer size is number of pixels 128^2, one hidden layer with 128 nodes and a softmax layer to output probabilities for the 7 classes in the dataset (2'114'432 trainable parameters).

Our softmax classifier is a one layer shallow neural network normalized with a softmax function output. It is similar to the Multilayer perceptron but has only a 7 neurons with each 16384 weights giving in total 114'688 trainable parameters.
.

4 Results

At **Figure 4** our results can be seen. The test score which is the percentage of correct classifications on the test data averaged over 5-fold cross-validation procedure.

It can be seen that the Convolutional Neural Network does not decrease in performance as fast as the other algorithms when we increase translational object variance. It is not a surprise that there is some decrease in all classifiers performance, because we keep the number of training images fixed as well as the size of the classifiers. When increasing the dimensions of variance we should also expect a lower performance. The interesting fact, seen at Figure 4, is the relative performance drop between the CNN and the other classifiers. It clearly shows that CNN can be trained to have translational invariance.

5 Conclusions

There are many ways to increase the problem size shown in this article towards more realistic scenarios for SAR ATR. We have created a simulated dataset that covers some many type of variance in target appearance. By controlling the amount translational variance of target alignment we have shown how CNN are superior to others machine learning algorithms in dealing with translational variance.

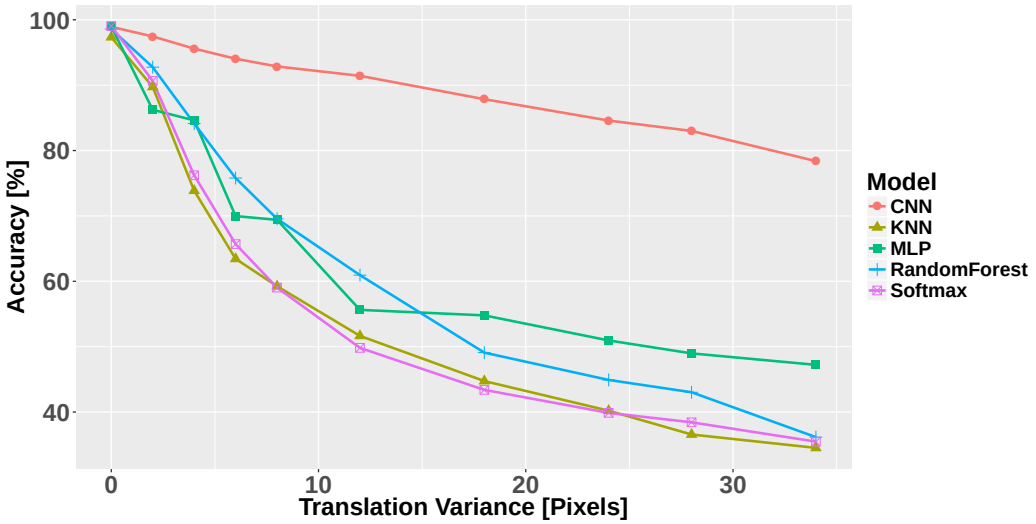


Figure 4: Test score plotted as a function of the percentage of randomly translational variance. Note that the first drop in performance happens around 3 pixels random translational variance

The impracticalities of collecting real SAR datasets that represent all natural variances of target representation for SAR ATR, are many. It is therefore very important to consider realistic problems when benchmarking algorithms on the sparse data available. We have shown that when considering the best algorithms for SAR ATR one must know the precision of placing targets consistently. Whether it is from a detection algorithm or manually extracted target patches, their displacement variability must be considered as even small inconsistent displacements can have big impact on the accuracy of a classifier. We have shown that some algorithms have a drastic decrease in performance when objects vary in position with as little as 3 pixels.

References

[1] Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[2] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. pages 1097–1105, 2012.

[3] David A. E. Morgan. Deep convolutional neural networks for ATR from SAR imagery. *Proc. SPIE*, 9475:94750F–94750F–13, 2015.

[4] Joseph A O'Sullivan, Michael D DeVore, Vikas Kedia, and Michael I Miller. SAR ATR performance using a conditionally gaussian model. *Aerospace and Electronic Systems, IEEE Transactions on*, 37(1):91–108, 2001.

[5] Umamahesh Srinivas, Vishal Monga, and Raghu G Raj. SAR automatic target recognition using discriminative graphical models. *Aerospace and Electronic Systems, IEEE Transactions on*, 50(1):591–606, 2014.

[6] Nitish Srivastava, Geoffrey E. Hinton, Alex Krizhevsky, Ilya Suskever, and Ruslan Slakhutdinov. Dropout: A simple way to prevent neural networks from overfitting. *The Journal of Machine Learning Research*.

[7] Steffen Wagner. Combination of convolutional feature extraction and support vector machines for radar ATR. In *Information Fusion (FUSION), 2014 17th International Conference on*, pages 1–6. IEEE, 2014.

[8] Qun Zhao and Jose C Principe. Support vector machines for SAR automatic target recognition. *Aerospace and Electronic Systems, IEEE Transactions on*, 37(2):643–654, 2001.

Paper B - Convolutional neural networks for SAR image segmentation

Convolutional Neural Networks for SAR Image Segmentation

David Malmgren-Hansen and Morten Nobel-Jørgensen
Technical University of Denmark
DTU Compute Department of Applied Mathematics and Computer Science
Richard Petersens Plads, building 324 DK-2800 Kgs. Lyngby
Phone: +45 45 25 34 22
Email and Homepage: dmal@dtu.dk and http://www.staff.dtu.dk/dmal

***Abstract*—Segmentation of Synthetic Aperture Radar (SAR) images has several uses, but it is a difficult task due to a number of properties related to SAR images.**
In this article we show how Convolutional Neural Networks (CNNs) can easily be trained for SAR image segmentation with good results. Besides this contribution we also suggest a new way to do pixel wise annotation of SAR images that replaces a human expert manual segmentation process, which is both slow and troublesome. Our method for annotation relies on 3D CAD models of objects and scene, and converts these to labels for all pixels in a SAR image.
Our algorithms are evaluated on the Moving and Stationary Target Acquisition and Recognition (MSTAR) dataset which was released by the Defence Advanced Research Projects Agency during the 1990s. The method is not restricted to the type of targets imaged in MSTAR but can easily be extended to any SAR data where prior information about scene geometries can be estimated.

I. Introduction

A lot of research has been done in image segmentation. The objective is typically to simplify image information so that different features or measures can be extracted and used for e.g. classification of objects. The same objectives are relevant when segmenting SAR images. For example objects' height above ground in a SAR image can be estimated from its shadow length, since the depression angle is inherently known, but only if the shadow can be well segmented.
Transferring image segmentation methods to SAR images is not easy. SAR images contain speckle that arises with scatterers complex summation within a resolution cell from coherent signals. This type of noise together with SAR sensors' representation of geometry makes it hard to rely on edge information in SAR image segmentation.
Relying purely on intensity information is also a problem since intensities are not unique for different areas in a SAR image. This is illustrated with the intensity density plots in Figure 1 for the type of areas concerned with the data we work on in this article.

Another problem in automatic segmentation is the difficulty of obtaining baseline annotations. We need these in order to test our algorithms, but they also enable supervised segmentation methods if parts of annotated data is used for optimizing prediction models. Manual expert annotation is a common way of measuring performance, but this is difficult in SAR images. G. J. Power and R. A. Weissenseel have shown in [1] that manual segmentation is a complicated process that benefits from their concept of supervisory control. They also show that manual segmentation suffers from inter- and intra operator variability.

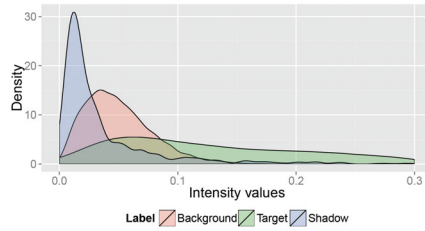


Fig. 1: The three density plots come from intensity values defined by the region of our annotation algorithm.

We propose a new method to create baseline segmentations to test other segmentation algorithms against. Our method uses 3D models of the known targets to create a labelled image with pixels annotated as target, shadow or background. Based on rasterization and further pixel remapping we are able to transform the 3D model into an approximated radar view. The software developed to create these segmentations along with source code will be publicly available[1]. Our software, named SARBake, is intended as a tool for research in MSTAR and similar datasets, but the concept could be extended to any SAR data where the objects and scene can be well modelled. On top of an object and landscape model, the algorithm requires knowledge about object orientation. It is therefore not suited as an automatic segmentation method in SAR image analysis. In the MSTAR dataset both object name and orientation are known for all images, which makes this data suitable for testing our algorithm. An example of a MSTAR image can be seen in Figure 2.

[1]Source code available on GitHub. Links and information can be found on the 1st author's homepage.

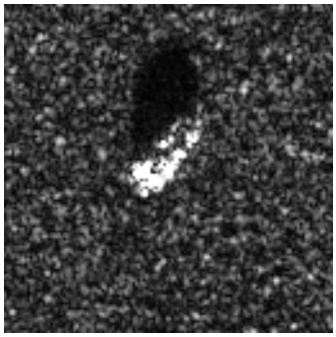


Fig. 2: SAR image from the public MSTAR database. The particular image shows a BTR60 armored personnel carrier.

In this article we will also investigate the performance of Convolutional Neural Networks (CNN) as a supervised segmentation approach.

CNNs have shown great results in computer vision lately as they are able to learn both hierarchical structured feature extractors and good classifiers in a single model optimization. Due to its great success in classification we have chosen to classify a patch around each pixel in the image according to the segment it belongs to. We have used a Deep Learning library for Python called Keras [2] that builds on top of the Python machine learning library Theano [3], [4]. Our method is further described in Sec. III

II. Method for Baseline Segmentations - SARBake

To create the baseline annotation, geometric properties of the target must be known. The MSTAR dataset is labelled with target vehicles, for example T62 tank or BTR60 armored personnel carrier, as well as viewing angle. CAD models of most vehicles are freely available from online communities and we use such given models as input to the method.

Simple rendering of a model does not provide information about the source of energy in the individual pixel intensities. This would require a simulation of recording and processing the SAR images, which is a complex and computationally expensive task. Our model produces a mask of the area illuminated by a radar wave when converted into SAR image coordinates.

Listed below are the assumptions made in our method,

1) With orthographic projection in the rendering phase, far-field radar viewing conditions are assumed.
2) The 3D models found for SAR targets are close approximations to actual targets imaged by the SAR sensor.
3) No target scattering properties are taken into account, i.e. we assume that surfaces illuminated by the radar wave belong to the given target.
4) Effects from a wide angle aperture of the radar are small and can be neglected, i.e. the SAR image can be approximated with a scan of 2D slices through a 3D object representation.

SARBake consists of two steps, model rendering and mapping from 3D points to 2D points.

A. Model Rendering

The model is rendered using an orthographic projection. When performing orthographic projection, view rays are parallel and the size of the projection does not depend on the distance from the virtual camera to the object. The pixels of the rendered image contain the distances to surface of the model as floating-point numbers. Since the resulting image contains depth from the virtual camera it is known as a depth map.

For each view configuration we render the target and the ground in two separate render calls. This allows us to compute the shadow cast by the model onto the ground, including the distance from the surface point to the ground point for each view ray. Using geometry to represent the ground allows us to easily extend the method to work on nonplanar ground surfaces.

This method could be implemented using a ray tracer, however it fits well with the hardware rasterizer found in modern graphics cards. Our implementation uses OpenGL to render the depth map. This is achieved by enabling hardware-accelerated z-buffering, which stores the minimum distance between the rendered surface and the image plane. After rendering, the values from the z-buffer are read and transformed as described in Sec. II-B.

The rendering speed depends on the graphics card, the complexity of the model and the resolution of the image plane. For a model of 500k triangles the rendering time is 14ms on a NVIDIA GeForce GT 650M 1024 MB graphics card.

B. Mapping to Radar Coordinates

The complete mapping of a 3D CAD model to pixel labels in a SAR image can be described as transforming points in a (x, y, z)-space to points on a (x, r)-plane. This is illustrated in Figure 3. As a SAR sensor records reflection intensities in range intervals, r, from its position along the trajectory x, (x, r) spans our image plane. From the model rendering process described in the previous section all points in (x, y, z)-space are transformed into points in $(x, r, \frac{z}{cos(\alpha)})$, where α is the depression angle of the radar.

In the depth map from the model rendering process we get an image where columns correspond to the displacement in the x-axis, and rows to the location on the transformed z-axis, $(\frac{z}{cos(\alpha)})$. The intensities of the output pixels represent the distance from the camera plane to the target. The next step is simply to eliminate the spatial location in rows. This is done by column wise re-sampling the depth intensities into a series of labelled pixels with "0" indicating background, "1" indicating target, and "2" indicating shadow. Figure 4 illustrates the concept of representing the different labels along a line perpendicular to the image plane. For the

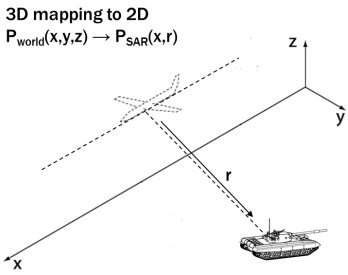


Fig. 3: Illustration of how a SAR sensor maps 3D real world points into a 2D plane expanded by the x displacement and the range from the radar.

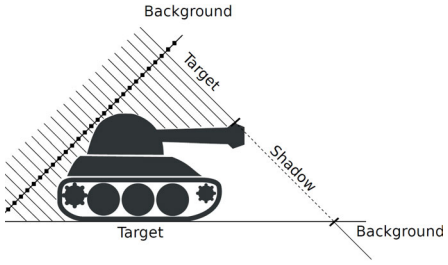


Fig. 4: Illustration of sampling along the r-axis.

cross-section in Figure 4, there is a distance where neither background nor target are present and this represents the shadow. The shadow distance could be calculated, but as described in the previous section, we render a ground plane and read the shadow distance from that.

If an exact target position is known when the SAR image is recorded, the segmentation can be positioned according to that. This is not the case with the MSTAR SAR images, and neither is a fixed offset of a model the solution to a good placement of our baseline segmentation. Instead we chose to perform a 2D cross-correlation between our target labels and the SAR image and take the position of the maximum as our placement. In Figure 5 a SARBake result can be seen.

III. Convolutional Neural Networks for Segmentation

CNNs for segmentation of both natural images and medical images have been investigated before, as shown in [5] and [6]. There are mainly two conceptually different approaches. One approach is to extract patches around every pixel in the image, and for each of these patches a label for the center pixel is predicted. This often works well, but for large images it is computationally costly. In the second approach,

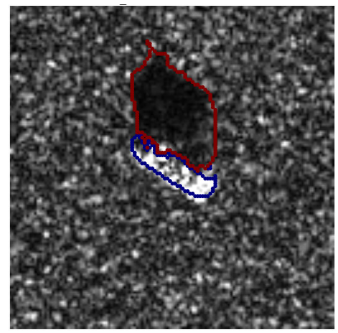


Fig. 5: SARBake baseline annotation of an MSTAR image. The vehicle imaged is a BTR60 armored personnel carrier.

shown in [6], the CNN model is rewritten into a function that can predict a full segmentation mask for each input image given. In this approach there is a compromise between reducing dimensionality so that many layers can learn complex hierarchical features and simultaneously preserve sharp edges in the segmentations.

Due to the SAR images in MSTAR being fairly small, most of them are 128x128 pixels, we have chosen to simply extract patches of size 33x33 pixels around every pixel in the image and predict a label for each of them. With a patch size of 33x33 pixels we ensure that enough context information from surrounding edges is included for the network to learn well describing features, but other patch sizes could as well have been chosen. We disregard the boundaries when extracting patches, thus we end up with a segmentation mask of the 96x96 for the center of the image.

In Figure 6, an illustration of parsing data through our model can be seen, as well as the network architecture. The design is inspired by early work in CNN research presented by LeCun et al. [7], but other model architectures can as well be explored for this task. Table I describes the individual layers in the model. It has 22,917 uniformly random initialized parameters, which must be considered a small CNN. However, considering the problem complexity and runtime for a whole image segmentation it seems reasonable to not choose a very large networks for this task.

Our loss function is the multiclass logloss function, also known as the categorical cross entropy loss function, and we optimize it with a stochastic gradient descent algorithm evaluated on batches of training data. The loss function can be seen in Eq. 1.

$$E(\mathbf{W}, \mathbf{D}_m) = \frac{1}{|\mathbf{D_m}|} \sum_{m=0}^{|\mathbf{D_m}|} \sum_{n=0}^{N} T_{m,n} log(y_{m,n}(x_m, \mathbf{W})) \quad (1)$$

where D_m is a minibatch of data and N is the number of labels (three in our case). $T_{m,n}$ is a binary vector of size N, where each element represents a label and the value represents

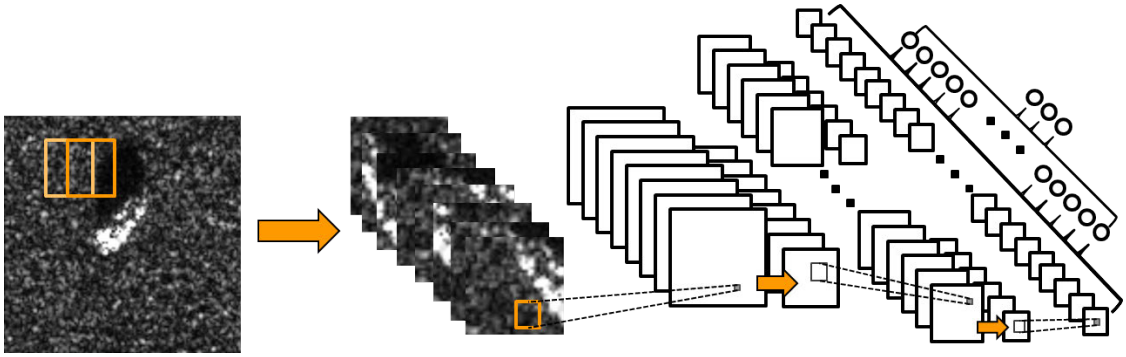


Fig. 6: Illustration of our Convolutional Neural Network model. The network output is probabilities of each class, target/shadow/background, for the center pixel of the input sample.

Type	Image Size	Feature Maps	Kernel Size
Input	33x33	1	-
Convolutional + ReLU	28x28	8	6x6
Max pooling	14x14	8	2x2
Convolutional + ReLU	10x10	18	5x5
Max Pooling	5x5	18	2x2
Convolutional + ReLU	3x3	100	3x3
Fully connected + ReLU	-	3	900
Softmax	-	3	-

TABLE I: Individual layers in the Convolutional Neural Network model. ReLU indicates that a Rectified Linear Unit activation function has been applied to the output.

Parameter	Value
α	0.9
β	$5 * 10^{-4}$
γ	$1 * 10^{-6}$
ϵ	0.0125

TABLE II: Training parameter values.

whether the label is true for the $m'th$ sample. $y_{m,n}$ is a predicted vector of probabilities for the $m'th$ sample, given the weights in $\mathbf{W}$.
When updating our weights we make use of momentum based learning, weight decay and learning rate decay. Our update function can be seen in Eq. 2.

$$\begin{aligned} \epsilon_{i+1} &= \epsilon_i * (1 - \gamma) \\ \mathbf{g}_{i+1} &= \alpha \mathbf{g}_i - \beta \epsilon_{i+1} \mathbf{W}_i - \epsilon_{i+1} \frac{dE}{d\mathbf{W}_i} \\ \mathbf{W}_{i+1} &= \mathbf{W}_i + \mathbf{g}_{i+1} \end{aligned} \tag{2}$$

where α is the momentum coefficient, β our weight decay, ϵ_{i+1} our learning rate at iteration $i+1$, and γ our learning rate decay. The values used for the training parameters are shown in Table II.

When performing the actual segmentation, patches around each pixel in the image are extracted and passed through the model, but at training time a balanced dataset of patches has been created. This has been done by selecting 100 random MSTAR images in the subset called "Mixed Targets", all recorded at 17° depression angle. This subset has seven different vehicles such as a bulldozer, a truck, personnel carriers, and tanks. All vehicles have been recorded with $<2°$ sampling of rotation angles. Since there are fewer target pixels than shadow- and background pixels in an MSTAR image, patch samples from the shadow and background class were removed until an equal amount of samples of each category was achieved. This yields approximately 100k samples in total, which has been divided in to a training and test set by a ratio of 70% / 30%.

At training 29,640 weight updates with a minibatch size of 128 samples were run. The whole training took 27 minutes on an NVIDIA TITAN Black graphics card, but the optimization converges fast, reaching an 11% test error after just 592 weight updates. After training we reached a test error of 5.81%.

IV. Results

The MSTAR Mixed Targets subset contains a total of 2,049 images of 7 vehicles recorded at 17° depression angle. CNN segmentation on all these images was performed, except the 100 images used for training.

We have chosen to report our performance measures in Dice scores, as it is a common measure in segmentation challenges. First shown in [8], Dice scores are defined as Eq. 3. A good property of the Dice score is that correct predictions are normalized with the mean area of the baseline annotation and the predicted segmentation. Since shadows often takes up more space in a SAR image taken from a low depression angle, than the target itself, Dice scores provides a more fair comparison between our performance on these two classes.

$$S_{Dice} = \frac{2|A \cap B|}{|A| + |B|} \tag{3}$$

where A is the segmentation result and B the baseline annotation to compare against.
Dice score density plots of our segmentation performance on

all 1949 test images can be seen in Figure 7.

A typical segmentation from our CNN model can be seen

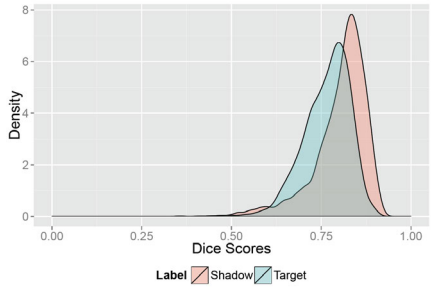
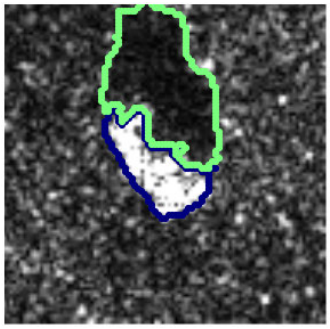
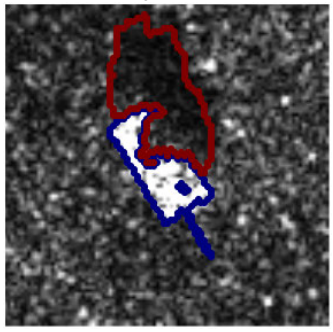


Fig. 7: Dice score density plots for target- and shadow segmentations. Target: $mean = 0.76$, $std = 0.065$. Shadow: $mean = 0.80$, $std = 0.074$

in Figure 8a with the corresponding SARBake annotation in Figure 8b. However, some results are corrupted by dark areas not located next to the target, as seen in Figure 9. The cause of this is likely to be shadows from objects outside of the image region.

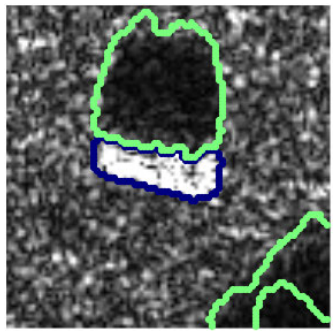


(a) CNN segmentation result.

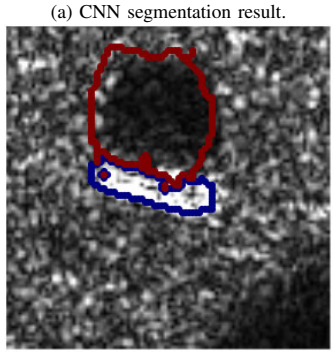


(b) SARBake segmentation result.

Fig. 8: T62 tank. 17° Depression angle, 329° azimuthal object rotation. Dice score $D_{target} = 0.82$, $D_{shadow} = 0.81$.



(a) CNN segmentation result.



(b) SARBake segmentation result.

Fig. 9: BTR60 armored personnel carrier. 17° Depression angle, 329° azimuthal object rotation. Dice score $D_{target} = 0.81$, $D_{shadow} = 0.71$.

A. *Re-evaluation*

The prediction error in the lower right corner in Figure 9a is clearly a result of inaccurate modelling in our baseline annotation. Regardless whether the dark area is due to a shadow cast from another object or from topographic variations, it is information we do not have for our annotation. It is therefore also relevant to test our prediction model excluding these areas. We have chosen to constrain our prediction so that pixels classified as shadow must lie behind a target pixel in every column of the image. This is done after prediction of a whole image, where pixels not fulfilling our constraint are set to be shadow. After this post processing, the result from Figure 9a is shown in Figure 10. The difference in Dice scores with and without this constraint on the prediction can be seen in Table III.

Furthermore, we evaluated our model on data recorded at 15° depression angle. At this angle there are an additional 1850 images. Despite not being part of the training data we achieve similar results at the 15° data, as seen in Table IV.

V. Conclusion

The result from our annotation process can only be qualitatively evaluated since no ground truth exist for the MSTAR

Dice score	Before	After
Mean	0.80	0.83
Standard deviation	0.075	0.065

TABLE III: Mean and standard deviations of shadow segmentation. Dice score before and after the post-processing algorithm that removes shadow labelled pixels if they are not located behind target labelled pixels.

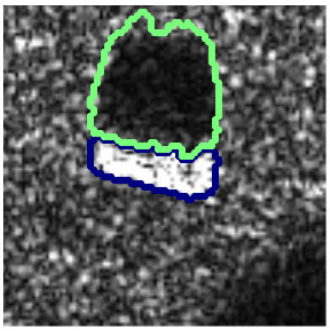


Fig. 10: The prediction from Figure 9a, where shadow pixels are constrained to be located behind target pixels.

Dice score	Target	Shadow
Mean	0.76	0.84
Standard deviation	0.078	0.066

TABLE IV: Mean and Standard Deviations of segmentation Dice score for both target and shadow when evaluated on data from 15 °depression angle.

images. The visual result seems very convincing, but the concept's drawback is that approximate CAD models of the target imaged must be obtained in order to achieve a good result. Our method is fast and consistent and, if 3D models were found for all objects in MSTAR, the total dataset of nearly 14000 images could be annotated in less than a day on a computer with a decent graphics card. For the seven vehicles in this dataset, adequate models were found by searching online CAD model sharing communities. We showed how a Convolutional Neural Network was able to learn segmentation of SAR images from our automatic annotation method, achieving low error rates. Our segmentation results were obtained without specific effort in fine tuning network architecture and training parameters. It could be relevant for future work to investigate the effect of different patch sizes as well as other model designs, but we have shown here that even with a simple structure good results can be achieved. Being able to predict connected segmentation masks that seem to rely on actual background features, this type of segmentation method seems very promising for SAR image data.

We will make all SARBake annotations used for this research available on the first author's website to encourage further research in the field.

REFERENCES

[1] G. J. Power and R. A. Weisenseel, "ATR subsystem performance measures using manual segmentation of SAR target chips," *Algorithms for Synthtic Aperture Radar Imagery VI*, vol. 3721, no. 1, pp. 11–24, 1999.

[2] F. Chollet, "Keras," https://github.com/fchollet/keras, 2015.

[3] F. Bastien, P. Lamblin, R. Pascanu, J. Bergstra, I. J. Goodfellow, A. Bergeron, N. Bouchard, and Y. Bengio, "Theano: new features and speed improvements," Deep Learning and Unsupervised Feature Learning NIPS 2012 Workshop, 2012.

[4] J. Bergstra, O. Breuleux, F. Bastien, P. Lamblin, R. Pascanu, G. Desjardins, J. Turian, D. Warde-Farley, and Y. Bengio, "Theano: a CPU and GPU math expression compiler," in *Proceedings of the Python for Scientific Computing Conference (SciPy)*, Jun. 2010, oral Presentation.

[5] D. C. Cireşan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Mitosis detection in breast cancer histology images with deep neural networks," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2013*. Springer, 2013, pp. 411–418.

[6] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," *arXiv preprint arXiv:1411.4038*, 2014.

[7] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, pp. 97–104, 2004.

[8] L. R. Dice, "Measures of the amount of ecologic association between species," *Ecology*, vol. 26, no. 3, pp. pp. 297–302, 1945. [Online]. Available: http://www.jstor.org/stable/1932409

Paper C - Improving SAR Automatic Target Recognition Models with Transfer Learning from Simulated Data

Improving SAR Automatic Target Recognition Models with Transfer Learning from Simulated Data

David Malmgren-Hansen, Anders Kusk, Jørgen Dall, Allan Aasbjerg Nielsen, Rasmus Engholm, and Henning Skriver

***Abstract*—Data driven classification algorithms have proven to do well for Automatic Target Recognition (ATR) in Synthetic Aperture Radar (SAR) data. Collecting datasets suitable for these algorithms is a challenge in itself as it is difficult and expensive. Due to the lack of labelled datasets with real SAR images of sufficient size, simulated data plays a big role in SAR ATR development, but the transferability of knowledge learned on simulated data to real data remains to be studied further.**

In this paper we show the first study of Transfer Learning between a simulated dataset and a set of real SAR images. The simulated dataset is obtained by adding a simulated object radar reflectivity to a terrain model of individual point scatters, prior to focusing. Our results show that a Convolutional Neural Network (Convnet) pre-trained on simulated data has a great advantage over a Convnet trained only on real data, especially when real data is sparse. The advantages of pre-training the models on simulated data show both in terms of faster convergence during the training phase and on the end accuracy when benchmarked on the MSTAR dataset. These results encourage SAR ATR development to continue the improvement of simulated datasets of greater size and complex scenarios in order to build robust algorithms for real life SAR ATR applications.

***Index Terms*—SAR ATR, Convolutional Neural Networks, Transfer Learning, SAR Image Simulation.**

I. Introduction

In Automatic Target Recognition (ATR) for Synthetic Aperture Radar (SAR) applications it is well known that lack of realistic and big labelled datasets is a challenge for the development of robust algorithms. For statistical studies of SAR ATR algorithm performance, it is important to have sufficient data. However, just as important is it to have a great variety of realistic scenarios. The latter should cover objects on different kind of backgrounds, (e.g. grass, road, gravel) but also scenes where objects are closely co-located, occluded by trees and similar challenging scenarios, dependent on its relevance to the ATR application at hand. Such datasets will be difficult or at least very expensive to collect for each individual SAR ATR application. Hence simulated data has great potential to improve this field.

The Moving and Stationary Target Acquisition and Recognition (MSTAR) dataset has been used for benchmarking algorithms since it was collected by AFRL[1] and DARPA[2] between 1995-1997. The vehicles in the dataset are densely sampled in azimuthal view angle but are only recorded at few depression angles and the backgrounds remain stationary throughout the dataset. Additionally, vehicles are centered in the image, and there are no confusing features in the background (trees, other vehicles, etc.). Due to these shortcomings it should not be considered as a random subset of operational interesting SAR ATR data [1]. Despite this, MSTAR is interesting since it can show an algorithm's robustness to the statistical properties of real SAR data. MSTAR can also reveal the algorithm's generalizability of features learned on one depression angle, to another, and thereby whether 2° difference in depression angle sampling is adequate in ATR development. Several data driven classification models have proven to do well on the MSTAR task, such as [2], [3], [4], [5], [6], [7]. In order to learn more about the algorithms' scalability to larger recognition tasks with more targets, or complex scenarios, the potential of simulated data is big.

Research on improving Convolutional Neural Networks (Convnets) by including simulated data dates back to 2004 [8] for natural images. Here training images of toy figures made the Convnet generalize to predictions on real images, when complex backgrounds were inserted in the training samples. In more recent work, [9], [10] it is suggested to use rendered images from 3D CAD models to help training Convnets in specific applications where labelled data is sparse. In [9] there is a specific focus on making the rendered images realistic. Instead of rendering complex backgrounds it is suggested to use real backgrounds and add rendered objects to them. Following such an approach for SAR data to enhance SAR ATR development is suggested several times in the literature, [11], [12], [13], but whereas realistic rendering is highly developed in the optical image domain, SAR simulation needs to deal with different challenges. These challenges include complex summation of reflections due to a coherent signal source, and edge diffraction due to the large wavelength compared to visible light. Several approaches to SAR image simulation have been suggested from very simple scatter models as the one used in [14] to sophisticated models that analyze the object and scene for geometric features that have significant scattering [15].

To open up for the use of simulated data in real SAR ATR applications, methods of exploiting it need to be explored further. This is both in terms of ensuring that the simulated

David Malmgren-Hansen and Allan Aasbjerg Nielsen are with the Department of Applied Mathematics and Computer Science at the Technical University of Denmark. e-mail: (see http://people.compute.dtu.dk/dmal/ or http://people.compute.dtu.dk/alan/)

Anders Kusk, Jørgen Dall and Henning Skriver are with National Space Institute at the Technical university of Denmark.

Rasmus Engholm is Senior Data Scientist at Terma A/S, Lystrup Denmark

[1] Air Force Research Laboratory

[2] Defense Advanced Research Projects Agency

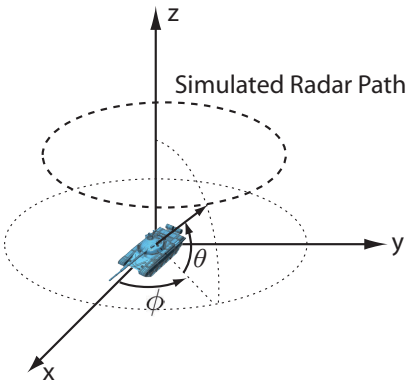


Fig. 1. Coordinate system used for simulated SAR images.

data is sufficiently realistic and to find practical ways of incorporating it into data driven classification algorithms. In [16] the combined use of simulated data and real data is explored in a study of pairwise binary classification tasks between SAR images with eight different ships. The exact ship models are carefully modelled in a CAD drawing and rendered with a SAR simulation tool. This study is not close to a real application in the sense that it only shows the separability between the pairs of ships, but reach an important conclusion of simulated data having a regularizing effect on Convnet training.

In this article we aim to explore the information that can be learned from a simulated SAR dataset which can be transferred to a real SAR data scenario. We do this by studying the accuracy of a Convnet pre-trained on the simulated data versus a Convnet trained with random initialized weights, i.e. Transfer Learning such as in [17]. With this approach objects in our simulated dataset, $\mathcal{D}_{sim}$, do not have to be geometric replicas of the real objects in our target dataset, $\mathcal{D}_{real}$, as long as they share image features. This is a great advantage for SAR ATR where exact target geometry rarely is known. In this way our method possesses both a practical way of incorporating simulated data into a SAR ATR application as it reveals whether the data is realistic enough to benefit a classification algorithm. We show that generic features learned on the simulated data benefit the model performance on real data. Our results show that training time can be reduced and end performance improved. In the experiments described in Section III we used simulated data from the tool introduced in [18] named SARSIM. This tool is described in Section II and the specific dataset used in this work will be publicly available for research purposes at [19]. The results from the Transfer Learning experiments are shown in Section IV and Section V concludes and summarizes our findings.

II. Data Simulation

To simulate realistic SAR images of an object, a model of the object radar reflectivity is required, as well as models of the surrounding terrain and the radar system. The object reflectivity is estimated using the commercially available CST[3] Microwave Studio Asymptotic Solver with a 3D CAD model as input. CST estimates the complex scattered electric field components over a range of frequencies, elevation (θ) angles and azimuth (ϕ) angles, using the geometry illustrated in Figure 1. The frequency sweep covers the simulated radar system bandwidth, and a full $360°$ sweep in azimuth is carried out for each desired elevation angle. Since the CST simulation results are sampled in frequency, the sampling increment, Δf, determines the alias-free scene extent (object size) in range, W_r, that can be simulated by $\Delta f < \frac{c}{2W_r}$, c being the speed of light. Likewise, the azimuth sampling increment, $\Delta\phi$ is determined not by the antenna beamwidth (as it is in a real system), but by the alias-free scene extent in azimuth, W_x, as $\Delta\phi < \frac{\lambda}{2W_x}$, with λ being the wavelength.

The geometry in Figure 1 assumes acquisition along a circular synthetic aperture. By using a time-domain back-projection focusing algorithm [20], however, the final output images can be provided on an arbitrary grid, e.g. a rectangular slant-range/azimuth grid. Compared to a linear aperture, simulation of a circular aperture yields a slightly different result due to varying incidence angles along the aperture. For an aperture corresponding to 10 cm resolution at X-band and nominal $\theta = 45°$, the difference is smaller than $0.2°$. This difference is independent of sensor altitude, and is smaller yet for lower depression angles. This means that even though linear aperture SAR images are desired, the circular aperture reflectivity simulation is likely a reasonable approximation. The output resolution is determined by the frequency bandwidth and azimuth angle interval used.

Terrain clutter is modelled by adding the object signal to the simulated backscatter of a set of individual point-like scatterers prior to SAR focusing. The scatterers are specified by (X, Y, Z)-position and complex reflectivity. For every aperture angle, the visibility of each scatterer is determined by casting rays from the sensor position to the scatterer and testing for intersection with the object CAD model. Scatterers for which the ray intersects the object are suppressed. For every visible scatterer, the range is calculated, and the simulated backscatter signal is generated by multiplying a flat spectrum with (a) the complex reflectivity of the scatterer, (b) a linear phase ramp that shifts the time response to have maximum at the calculated range, and (c) the two-way propagation phase. The summed contributions of all visible scatterers are added to the object signal. Simulated thermal noise is then added to every sample based on the radar equation and the simulated SAR system parameters. Finally, the signal is focused in range by an inverse FFT (with Taylor weighting for sidelobe suppression), and azimuth focusing is done using the back-projection algorithm. The approach above ensures that the clutter and noise signals are subjected to the same SAR processing as the object signal, and also models shadowing and partial visibility along the aperture correctly. Arbitrary scatterers can be used for the clutter simulation; currently, homogeneous clutter is simulated by a point scatterer at every

[3]CST - Computer Simulation Technology, Dassault Systemes

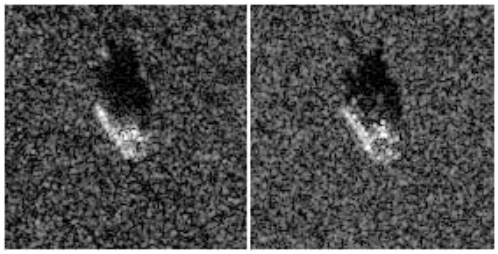


Fig. 2. Real vs. simulated images of T-72 at 15° depression and 148° azimuth. Left MSTAR actual SAR image, Right SARSIM simulated image.

Fig. 3. MSTAR T-72 tank and T-72 CAD model used for simulated image.

output grid point, with complex reflectivity modelled by a complex zero-mean Gaussian process with identical real and imaginary part variance s^2:

$$s^2 = (\sigma^0 \delta_a \delta_r)/2 \cos\theta \tag{1}$$

where σ^0 is the terrain reflectivity, and δ_a and δ_r are the range and azimuth pixel spacings. Typical values of σ^0 for different terrain types are available in [21].

Figure 2 shows a real MSTAR SAR image of a T-72 tank and the corresponding simulated image generated with the SARSIM tool, using a T-72 CAD model and MSTAR radar parameters, assuming a grassy terrain. A photo of the tank, as well as a rendering of the CAD model are shown in Figure 3.

III. Experiments

The experiments explained in this section are intended to investigate the transferability of information learned in a parametric model. Let us consider a classification task on a dataset, $\mathcal{D}$, solved with a Convnet model, $y(\mathbf{x}, \mathbf{w})$, with input $\mathbf{x} \in \mathcal{D}$. The model parameters or weights, $\mathbf{w}$, are initialized following the approach in [22] with a uniform distribution in the interval,

$$\mathbf{w} \sim U[-\frac{\sqrt{6}}{\sqrt{n_l + n_{l+1}}}, \frac{\sqrt{6}}{\sqrt{n_l + n_{l+1}}}], \quad for\ l = 0, ..., L-1 \tag{2}$$

with L being the number of layers in our model, and n_l being the number of incoming nodes in the l'th layer. For a multi class classification task, the maximum likelihood optimization of our model will be given with the categorical cross entropy error function if the output of the network is a softmax function, [23]. The error function is given by,

$$E(\mathbf{w}) = -\sum_{n=1}^{N} \mathbf{t}_n \ln(y(\mathbf{x}_n, \mathbf{w})), \tag{3}$$

with $\mathbf{t}_n$ being a "one-hot" vector encoding a one for the correct class and zeros otherwise given the n'th image. $y()$ has an output representing the probability of each k-classes in the given classification task. The error function can then be minimized with a stochastic gradient descent (SGD) scheme. In our analysis we use a version of SGD where the gradient for each iteration is divided with a root-mean-square weighted sum of the weights and previous gradients. This method called RMSprop [24], helps ensure a stable learning curve by effectively adjusting the step size of the weight updates.

In order to explain the amount of information that can be transferred from a simulated dataset, $\mathcal{D}_{sim}$, to a real dataset, $\mathcal{D}_{real}$, we investigate the error, $E(\mathbf{w})$, during the minimization process of Equation (3) with $\mathbf{x}_n \in \mathcal{D}_{real}$, for the following two scenarios.

1) $\mathbf{w}$ being initialized according to Equation (2).
2) $\mathbf{w}$ being initialized from $\underset{w}{\operatorname{argmin}}\ E(\mathbf{w}, \mathbf{x}),\ for\ \mathbf{x} \in \mathcal{D}_{sim}$.

Additionally, we can consider random subsets of $\mathcal{D}_{real}$ in order to study the relationship with dataset sparsity. The amount of data needed for training a model to SAR ATR is critical in operational scenarios, since it can be difficult to obtain dense datasets of the relevant objects. It should be emphasized that in our setup the classification task on $\mathcal{D}_{sim}$ and $\mathcal{D}_{real}$ can differ in terms of objects and complexity. Since the number of classes in $\mathcal{D}_{sim}$ and $\mathcal{D}_{real}$ in our experiment differ, we re-initialize the last layer with random weights according to Equation (2) in both scenarios. The two datasets used in the experiments are described in Section III-A.

The Convnet used in our experiments has 412,028 parameters. Layers and input sizes of the Convnet can be seen in Table I. We run our optimization by mini batch gradient descent with a batch size of 128 images. We further add L_2-norm weight decay to our update scheme.

Several approaches to Transfer Learning are reported in literature where only the last layer is fine tuned on a new dataset or where early layers are fine tuned with reduced learning rate. We will fine tune all parameters in the Convnet with equal learning rate as this was experimentally shown as the best procedure in [17].

On the standard MSTAR 10-way classification task with randomly initialized weights the model presented in Table I reaches 93.2% accuracy on the test set.

A. Datasets

The two datasets used in the experiments are described below.

1) MSTAR: MSTAR is a set of real SAR images of military vehicles. The subset often used for testing classification algorithms contains 10 vehicle types recorded at 15° and

TABLE I
CONVNET MODEL USED IN EXPERIMENTS. THE OUTPUT SIZE DENOTES *(number of rows x number of columns x number of nodes)* IN EACH LAYER. RECTIFIED LINEAR UNIT IS AN ACTIVATION FUNCTION DEFINED BY $f(x) = max(0, x)$.

Layer Type	**Layer Output size**	**Kernel size**	**comment**
Input	128x128x1	-	-
Convolutional	128x128x12	5x5	ReLU activation
Maxpooling	42x42x12	3x3	
Convolutional	42x42x36	5x5	ReLU activation
Maxpooling	21x21x36	2x2	
Convolutional	17x17x72	5x5	ReLU activation
Maxpooling	8x8x72	2x2	
Fully connected	56	1x4608	ReLU activation
Fully connected	10	1x56	Softmax activation

17° depression angles. The vehicles are densely sampled in azimuthal rotation with an image for each 1° - 2° depending on the object. The training set contains the ten vehicles at 17° and the test set contains the 15° samples.

2) SARSIM: The dataset consists of simulated SAR images of vehicles and is simulated according to the description in Section II. Fourteen vehicle CAD models have been downloaded from the web and each one is simulated for every two degree azimuthal object rotation at seven different depression angles (15°, 17°, 25°, 30°, 35°, 40° and 45°). The objects belong to seven different categories with two objects in each category (truck, car, motorbike, bus, tank, bulldozer and pick-up). Each object has been simulated with three different statistically generated background clutter types, corresponding to grass, roads and a mean of the two.

In our experiments the dataset is split randomly with 90% for training and 10% for testing. The effect on end performance by splitting the data in different ways (e.g. by depression angle, object model instance etc.) is not in the scope of this article to explore, but could be relevant for future studies. With the random split of 90%/10% we achieve a high accuracy on the test set (>99%). This is probably due to the very dense sampling of objects in different view angles.

IV. RESULTS

For each of the Convnet training scenarios on MSTAR, we consider five random subsets of the full dataset, (20%, 40%, 60%, 80% and 100%). In Figure 4 the accuracy on the test set is shown for each epoch, with epoch being the number of batch iterations corresponding to all images in the dataset being processed.

It is clearly seen that pre-training on simulated data makes the network converge faster on the real data set. This can be a great advantage in operational SAR ATR scenarios where problems can involve more vehicles than present in the MSTAR and where new vehicles are added regularly. Additionally, MSTAR lacks the variance of objects on different backgrounds, which makes real problems require even bigger datasets to fulfil operational requirements.

The difference in the end performance between the randomly initialized model and the pre-trained model seems to diminish with increasing dataset sizes. To illustrate this, Figure 5 shows a bar-plot of the end performance for each experiment.

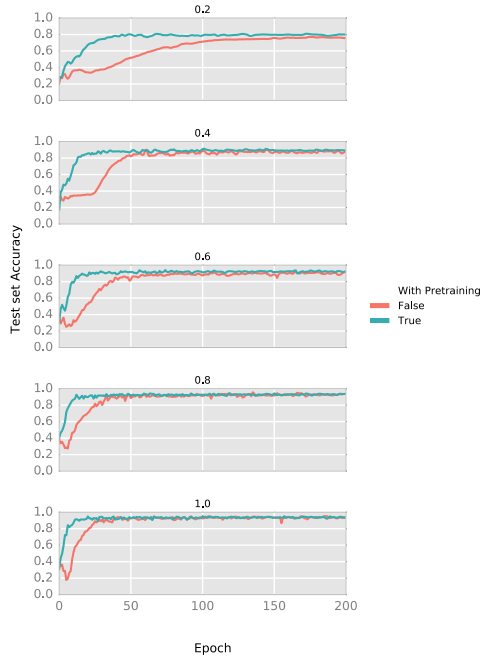


Fig. 4. Error on test dataset measured for each epoch during the training process. Different sub-plots show the fraction of the training data that was used.

Fig. 5. End test performance averaged over the last 10 epochs to even out test error variance due to training set/test set bias.

The end performance is reported as the average over the last ten epochs in order to reduce the variance on the test accuracy. This variance is likely due to biases between test and training sets and since the MSTAR problem splits between test and training on the depression angle, there is a significant bias between them. This variance was also reported in [2].

The difference in performance with increasing percentage of data included in the training is likely due to the very dense sampling in azimuth angles and lack of background clutter

variance in MSTAR. In operational scenarios it is very unlikely that such a dense sampling can be obtained for all objects of concern. Figure 5 shows that there is a great potential for increased performance with pre-training on simulated data when only sparsely sampled datasets are available.

V. Conclusion

The advances in simulation of SAR images lead towards practical data driven classification algorithms being useful in SAR ATR. We have shown that the simulated SAR images do in fact contain valid information that is useful for real SAR data ATR problems. The simulated data used in this work is improving both training convergence and the end performance of the classifier.

Transfer Learning provides a favourable scheme in utilizing simulated data. It enables the use of the many detailed CAD models freely available online to learn generic features that can be transferred to real SAR images in ATR applications. As detailed CAD models might be hard to obtain of all objects in a given operational classification task, Transfer Learning might be necessary to alleviate the lack of sufficient amounts of real SAR data.

The proposed method can as well be useful for other remote sensing problems, such as land cover classification and detection of objects in cities. In the proposed simulation algorithm it is assumed that object and ground interaction effects are negligible by separately simulating each of them. When comparing with the MSTAR images this assumption seemed reasonable, though it might be different for other radar frequencies and different environments. By proposing Transfer Learning between the simulated and real data it is likely that generic features can be learned although some simplifications are made in the simulation process. A study of which parts of the simulation are important in order to make Convnets generalize between the two data domains is an interesting subject for future work.

For the future improvement of SAR ATR, bigger and more realistic datasets need to be gathered. Including more complex scenes such as co-located vehicles, occlusions by trees and confusing objects such as buildings, will be the next step in the development. The performance of algorithms on a greater variety of object types will also shed light on data driven classification methods' robustness in operational SAR ATR applications. It must be expected that in operationally interesting applications, algorithms should deal with far more vehicle types compared to MSTAR and SARSIM. As shown in our experiments simulated data may play an important role in these studies.

Acknowledgement

The authors would like to thank Terma A/S for funding the development of SARSIM as well as the studies carried out here.

References

[1] T. D. Ross, S. W. Worrell, V. J. Velten, J. C. Mossing, and M. L. Bryant, "Standard SAR ATR evaluation experiments using the MSTAR public release data set," in *Aerospace/Defense Sensing and Controls*. International Society for Optics and Photonics, 1998, pp. 566–573.

[2] D. A. E. Morgan, "Deep convolutional neural networks for ATR from SAR imagery," *Proc. SPIE*, vol. 9475, pp. 94 750F–94 750F–13, 2015. [Online]. Available: http://dx.doi.org/10.1117/12.2176558

[3] J. A. O'Sullivan, M. D. DeVore, V. Kedia, and M. I. Miller, "SAR ATR performance using a conditionally gaussian model," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 1, pp. 91–108, 2001.

[4] U. Srinivas, V. Monga, and R. G. Raj, "SAR automatic target recognition using discriminative graphical models," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 50, no. 1, pp. 591–606, 2014.

[5] Y. Sun, Z. Liu, S. Todorovic, and J. Li, "Adaptive boosting for SAR automatic target recognition," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 43, no. 1, 2007.

[6] S. Wagner, "Combination of convolutional feature extraction and support vector machines for radar ATR," in *Information Fusion (FUSION), 2014 17th International Conference on*. IEEE, 2014, pp. 1–6.

[7] Q. Zhao and J. C. Principe, "Support vector machines for SAR automatic target recognition," *IEEE Transactions on Aerospace and Electronic Systems*, vol. 37, no. 2, pp. 643–654, 2001.

[8] Y. LeCun, F. J. Huang, and L. Bottou, "Learning methods for generic object recognition with invariance to pose and lighting," in *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, vol. 2. IEEE, 2004, pp. II–104.

[9] X. Peng, B. Sun, K. Ali, and K. Saenko, "Learning deep object detectors from 3d models," in *Proceedings of the IEEE International Conference on Computer Vision*, 2015, pp. 1278–1286.

[10] A. Shrivastava, T. Pfister, O. Tuzel, J. Susskind, W. Wang, and R. Webb, "Learning from simulated and unsupervised images through adversarial training," *arXiv preprint arXiv:1612.07828*, 2016.

[11] K. Tang, X. Sun, H. Sun, and H. Wang, "A geometrical-based simulator for target recognition in high-resolution SAR images," *IEEE Geoscience and Remote Sensing Letters*, vol. 9, no. 5, pp. 958–962, 2012.

[12] T. Balz, H. Hammer, and S. Auer, "Potentials and limitations of SAR image simulators–a comparative study of three simulation approaches," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 101, pp. 102–109, 2015.

[13] R. Sharma and N. S. Subotic, "Construction of hybrid templates from collected and simulated data for SAR ATR algorithms," in *Aerospace/Defense Sensing and Controls*. International Society for Optics and Photonics, 1998, pp. 480–486.

[14] D. Malmgren-Hansen, R. Engholm, and M. O. Pedersen, "Training convolutional neural networks for translational invariance on SAR ATR," in *EUSAR 2016: 11th European Conference on Synthetic Aperture Radar, Proceedings of*. VDE, 2016, pp. 1–4.

[15] N. Ødegaard, A. Knapskog, C. Cochin, and B. Delahaye, "Comparison of real and simulated SAR imagery of ships for use in ATR," in *SPIE Defense, Security, and Sensing*. International Society for Optics and Photonics, 2010, pp. 76 990O–76 990O.

[16] N. Ødegaard, A. O. Knapskog, C. Cochin, and J.-C. Louvigne, "Classification of ships using real and simulated data in a convolutional neural network," in *Radar Conference (RadarConf), 2016 IEEE*. IEEE, 2016, pp. 1–6.

[17] J. Yosinski, J. Clune, Y. Bengio, and H. Lipson, "How transferable are features in deep neural networks?" in *Advances in neural information processing systems*, 2014, pp. 3320–3328.

[18] A. Kusk, A. Abulaitijiang, and J. Dall, "Synthetic SAR image generation using sensor, terrain and target models," in *EUSAR 2016: 11th European Conference on Synthetic Aperture Radar, Proceedings of*. VDE, 2016, pp. 1–5.

[19] A. Kusk, J. Dall, and H. Skriver, "Simulated SAR data of vehicles on a background [data set]." http://doi.org/10.5281/zenodo.573750, 2017.

[20] Ulander, L.M.H. and Hellsten, H. and Stenstrom, G., "Synthetic-aperture radar processing using fast factorized back-projection," *Aerospace and Electronic Systems, IEEE Transactions on*, vol. 39, no. 3, pp. 760–776, 2003.

[21] F. Ulaby and M. Dobson, *Handbook of radar scattering statistics for terrain*, ser. The Artech House remote sensing library. Artech House, 1989.

[22] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks." in *Aistats*, vol. 9, 2010, pp. 249–256.

[23] C. Bishop, *Pattern Recognition and Machine Learning*, 8th ed., ser. Princeton Legacy Library. Springer, 2006.

[24] G. Hinton, N. Srivastava, and K. Swersky, "Lecture 6a overview of mini–batch gradient descent," *Coursera Lecture slides https://class. coursera. org/neuralnets-2012-001/lecture,[Online*, 2012.

Paper D - Spatial noise-aware temperature retrieval from infrared sounder data

SPATIAL NOISE-AWARE TEMPERATURE RETRIEVAL FROM INFRARED SOUNDER DATA

David Malmgren-Hansen[†], *Valero Laparra*[‡], *Allan Aasbjerg Nielsen*[†] *and Gustau Camps-Valls*[‡]

[†]Department of Applied Mathematics and Computer Science, Technical University of Denmark, Denmark
[‡]Image Processing Laboratory (IPL), Universitat de València, Spain

ABSTRACT

In this paper we present a combined strategy for the retrieval of atmospheric profiles from infrared sounders. The approach considers the spatial information and a noise-dependent dimensionality reduction approach. The extracted features are fed into a canonical linear regression. We compare Principal Component Analysis (PCA) and Minimum Noise Fraction (MNF) for dimensionality reduction, and study the compactness and information content of the extracted features. Assessment of the results is done on a big dataset covering many spatial and temporal situations. PCA is widely used for these purposes but our analysis shows that one can gain significant improvements of the error rates when using MNF instead. In our analysis we also investigate the relationship between error rate improvements when including more spectral and spatial components in the regression model, aiming to uncover the trade-off between model complexity and error rates.

***Index Terms*—** Infrared Atmospheric Sounding Interferometer (IASI), Minimum Noise Fractions, Principal Component Analysis (PCA), Statistical retrieval.

1. INTRODUCTION

"Perfection is achieved not when there is nothing more to add, but when there is nothing more to take away."
— Antoine de Saint-Exupry: Terre des hommes.

Temperature and water vapour atmospheric profiles are essential meteorological parameters for weather forecasting and atmospheric chemistry studies. Observations from high spectral resolution infrared sounding instruments on board of satellites provide for retrieval of such profiles. However, it is not trivial to retrieve the full information content from radiation measurements; accordingly, improved retrieval algorithms are desirable to achieve optimal performance for existing and future infrared sounding instrumentation.

EUMETSAT, NOAAA, NASA and other agencies are continuously developing product processing facilities to obtain L2 atmospheric profile products from infrared hyperspectral radiance instruments, such as IASI. One of the retrieval techniques commonly used in L2 processing is based on linear regression, which is a valuable and very computationally efficient method. It consists of performing a canonical least squares linear regression on top of the data projected onto the first principal components or Empirical Orthogonal Functions (EOF) –known in statistics as PCA– of the measured brightness temperature spectra (or radiances) and the atmospheric state parameters. To further improve the results of this scheme for retrieval, nonlinear statistical retrieval methods, as well as nonlinear pre-processing methods [1], can be applied as an efficient alternative to more costly optimal estimation (OE) schemes. These methods have proven to be valid in retrieval of temperature, dew point temperature (humidity), and ozone atmospheric profiles when the original data are used [2, 3]. However, they are costly to train and do not consider spatial correlation between radiances neither the noise information.

Recently, in [4], a high improvement on the performance of retrieval methods was reported when applying standard compression algorithms to the images. Although this result may appear counter-intuitive since compression implies reduction on the amount of information in the images, a certain level of compression is actually useful because: 1) compression removes information but also noise, and it could be useful to remove the components with low signal-to-noise ratio.; and 2) spatial compression introduces in a simple way information about the neighboring pixels. The use of Minimum Noise Fractions (MNF) employed here is a simpler and more mathematically elegant way to take advantage of both properties simultaneously. MNF is specifically designed to sort components according to the signal-to-noise ratio (SNR) [5]. The way we apply MNF here also enforces the inclusion of spatial information as noise is estimated by the residuals of fitting a quadratic surface locally. In this work we compare the effect of using PCA or MNF when retrieving temperature profiles using IASI data. We will show that MNF is better suited for this task. Moreover since PCA and MNF are both linear and unsupervised transformations, using MNF do not introduce any modification in the data processing pipeline.

The remainder of the work is organized as follows. Section §2 describes the data set collected and the pre-processing for dimensionality reduction and spatial filtering. Section §3 reviews the two decomposition methods used in the work. Section §4 gives empirical evidence of performance of the proposed scheme for spatial, noise-aware retrieval of atmospheric parameters. We conclude in §5 with some remarks and outline for the further work.

The research was funded by the European Research Council (ERC) under the ERC-CoG-2014 SEDAL project (grant agreement 647423), and the Spanish Ministry of Economy and Competitiveness (MINECO) through the projects TIN2015-64210-R and TEC2016-77741-R.

2. DATA DESCRIPTION

The Infrared Atmospheric Sounding Interferometer (IASI) data are point measurements of approximately 25 km diameter with 8461 spectral components, ranging in the infrared emission spectra from 645 to 2760 cm^{-1} with 0.25 cm^{-1} resolution. The dataset collected for this paper consists of 4 consecutive orbits from august 2013 of which the first three are used for training the regression model and the last is used for testing.

In our problem we follow the same scheme proposed in [4]. First we remove certain bands from the spectrum that do not contains useful information for retrieval reducing the data to 4699 spectral components. Although the longitudinal distance between acquisition points increases towards equator we can reshape each orbit into a rectangular grid of 1530×60 elements. By doing so, data can be treated as an image, taking advantage of spatial relations. The dimensionality reduction transformations are calculated on the training set and applied to both the training and testing datasets.

3. DECOMPOSITION METHODS

In our analysis we consider two orthogonal transformations, PCA [6] and MNF [5]. Notationally, given an observation data matrix $\mathbf{X} \in \mathbb{R}^{n \times d}$ with n pixels of d dimensions, we aim to find a transformation to a lower dimensional representation, $d' < d$, such that the projected data preserves most of the 'information' of the input. Solutions offered by both PCA and MNF are found by solving an eigenvalue problem but where the PCA finds a solution with eigenvectors in the columns of $\mathbf{W} \in \mathbb{R}^{d \times d'}$ in direction of maximum variance, the MNF looks for the eigenvectors that minimize the noise fraction, or equivalently maximizes the signal-to-noise ratio [7, 8]:

$$\begin{aligned} \text{PCA} : \mathbf{W}_* &= \arg\max_{\mathbf{W}} \left\{ \text{Tr} \left(\frac{\mathbf{W}^\top \mathbf{X}^\top \mathbf{X} \mathbf{W}}{\mathbf{W}^\top \mathbf{W}} \right) \right\} \\ \text{MNF} : \mathbf{W}_* &= \arg\max_{\mathbf{W}} \left\{ \text{Tr} \left(\frac{\mathbf{W}^\top \mathbf{X}^\top \mathbf{X} \mathbf{W}}{\mathbf{W}^\top \mathbf{X}_N^\top \mathbf{X}_N \mathbf{W}} \right) \right\}, \end{aligned} \tag{1}$$

where $\mathbf{X}$ is our data matrix with each row representing a sample of a infrared spectrum and with columns corresponding to the number of spectral components. $\mathbf{X}_N$ is the corresponding noise estimation of each sample in $\mathbf{X}$. The resulting set of vectors from the PCA decomposition are orthogonal as opposed to the MNF solution which obtains orthogonality with respect to the noise covariance.

If the noise covariance matrix is known, it can be used in the MNF estimation. Often it is not the case and it has to be estimated from data. Common ways to do noise estimation in image analysis include local mean subtraction, or taking the residuals from a plane or paraboloid fit on every pixel position in the image. We follow the latter approach for our analysis with a 3×3 paraboloid residual kernel implemented as a filtering operation [9].

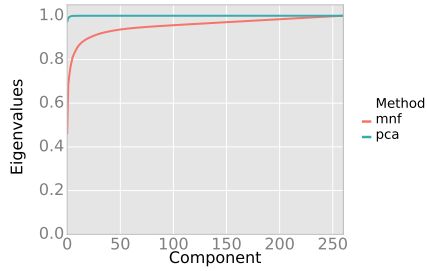


Fig. 1. Cummulated normalized eigenvalues.

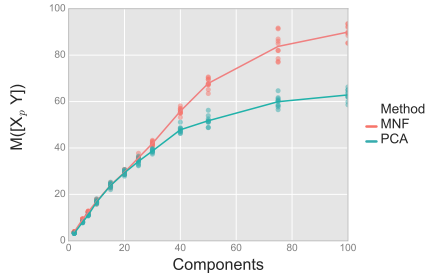


Fig. 2. Multi-information between the input and output components for PCA and MNF. Ten realizations have been made for each method while including different amount of spectral components. Lines denote the mean trend of the results.

The cumulative and normalized eigenvalues for both methods are shown in Fig. 1. For PCA they represent the percentage of explained total variance and it is seen that 99% explained variance is obtained within the first 5 components. For the MNF, eigenvalues represent the signal fraction for each component [5] and less than 80% signal fraction is obtained from the first 5 components. Although this could be seen as an disadvantage one has to take into account that PCA might keep the noise information too. Therefore how the eigenvalues relates to the information necessary to predict the temperature profiles $\mathbf{Y} \in \mathbb{R}^o$ is less straight forward to estimate. We analyze in Fig. 2 this relation by using the concept of *multiinformation* [10] (also known as *total correlation*). We show the amount of multiinformation, i.e. shared information, between the projected inputs $\mathbf{X}_p = \mathbf{XW}$ and the outputs $\mathbf{Y}$ using different amount of input components for each decomposition method. These values have been computed using RBIG method ([11]). We have followed similar procedure as in [12] where the amount of information contained by spatial and spectral components was analyzed for several sensor configurations. In this case, we are analyzing only the spectral information, yet including the variable to predict $\mathbf{Y}$ (temperature profiles). Fig. 2 shows the multiinformation results for PCA and MNF. Although it also includes the redundant information of the inputs, this measure can be

seen as an approximation of the information of the output that we can be obtained from the input. Note that even that MNF is not specifically designed to maximize this information, the multiinformation is bigger for MNF when using the same number of input components than for PCA. We will see in the experiments section how this behavior gives raise to improved retrieval performance.

As suggested in [4], to improve the retrieval performance it is important to remove noise from the data and to include spatial information. Fig. 3 illustrates the ability of MNF to do so. We show half orbit of data from the test set projected onto each of the 50 first components from the PCA (top row) and MNF (bottom row) decomposition. It is clear from this figure that MNF obtains smoother are less noisy projections than PCA. For instance component 38 from the PCA projection seem to contain less structure than the three following projections. This indicates that some noise components in the data have higher variance than other signal components. This behaviour repeats above the first 50 PCA components, whereas the MNF projections represents spatially smooth information in early components and gradually increase to finer details for higher components.

4. EXPERIMENTAL RESULTS

The goal of our experiments is two-fold, first to compare the effect of using PCA or MNF in the retrievals, and furthermore to uncover the trade-off between prediction performance and the number of spectral components included for each method. Dimensionality reduction is important to limit the computational load but choosing the appropriate number of components to keep is less straightforward. A lower computational load can be traded for larger amounts of training data so overfitting is prevented. Alternatively the lower number of data dimensions can enable the use of computationally heavier nonlinear models such as Kernel Ridge Regression, which has been shown to improve performance for retrieval in infrared sounder data [4].

As well as the influence of spectral sampling in temperature profile modelling we include experiments for different sizes of pixel neighborhood sampling as studies suggest this can be beneficial [12]. This means that we model the temperature profile of one sample in the IASI data from the sample plus a neighborhood of samples around it. For quadratic neighborhoods the increase of size will also lead to quadratic increase in computational load and it is therefore relevant to limit it.

In Fig. 4 the results from our experiments are shown. It is seen that the RMSE improvements converges after approximately 125 spectral components. The results also show that there is a significant improvement including neighborhood pixels in the modelling of temperature profiles, but that the improvement decreases going towards larger neighborhood sizes. Figure 5 shows the resulting RMSE over the temperature profile for using 175 spectral components in the Linear regression model. Our analysis suggests to use between $125-175$ spectral components from a MNF decomposition and a pixel neighborhood sampling size of 3×3 or 5×5 when performing Linear Regression on this type of data.

5. CONCLUSIONS

This paper showed that using MNF is a simple and mathematically elegant way of removing the noise in the signal and at the same time taking into account spatial information. These two properties have been suggested previously as an important point when dealing with this particular data [4]. Both effects can be observed in Fig. 3, the selected features by the MNF are less noisy and spatially softer than the ones found by PCA. We want to stress the fact that substituting PCA by MNF would not change the processing pipeline. PCA and MNF are both linear transformations so only the values of the projecting vectors should be changed. Moreover, unlike other solutions as PLS [13], PCA and MNF are unsupervised methods, i.e. are not fitted for predicting an specific variable. Therefore, although we here show the results for a particular variable (i.e. temperature), it is expected that the improvement would be consistent for the retrieval of other variables.

6. REFERENCES

[1] J. Arenas-Garca, K.B. Petersen, G. Camps-Valls, and L.K. Hansen, "Kernel multivariate analysis framework for supervised subspace learning: A tutorial on linear and kernel multivariate methods," *IEEE Signal Processing Magazine*, vol. 30, no. 4, pp. 16–29, 2013.

[2] G. Camps-Valls, J. Munoz-Mari, L. Gomez-Chova, L. Guanter, and X. Calbet, "Nonlinear statistical retrieval of atmospheric profiles from MetOp-IASI and MTG-IRS infrared sounding data," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 50, no. 5, pp. 1759–1769, 2012.

[3] G. Camps-Valls, V. Laparra, J. Muñoz-Marí, L. Gómez-Chova, and X. Calbet, "Kernel-based retrieval of atmospheric profiles from IASI data," in *IEEE Proc. IGARSS 11*, Jul 2011, pp. 2813–2816.

[4] J. García Sobrino, Serra-Sagristà, V. Laparra, X. Calbet, and G. Camps-Valls, "Statistical atmospheric parameter retrieval largely benefits from spatial-spectral image compression," *IEEE Transactions on Geoscience and Remote Sensing*, 2017.

[5] Andrew A Green, Mark Berman, Paul Switzer, and Maurice D Craig, "A transformation for ordering multispectral data in terms of image quality with implications for noise removal," *IEEE Transactions on geoscience and remote sensing*, vol. 26, no. 1, pp. 65–74, 1988.

[6] Harold Hotelling, "Analysis of a complex of statistical variables into principal components.," *Journal of educational psychology*, vol. 24, no. 6, pp. 417, 1933.

[7] Allan Aasbjerg Nielsen, "Kernel maximum autocorrelation factor and minimum noise fraction transformations," *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 612–624, 2011.

[8] Luis Gómez-Chova, Allan A Nielsen, and Gustavo Camps-Valls, "Explicit signal to noise ratio in reproducing kernel hilbert spaces," in *Geoscience and Remote Sensing Symposium (IGARSS), 2011 IEEE International*. IEEE, 2011, pp. 3570–3573.

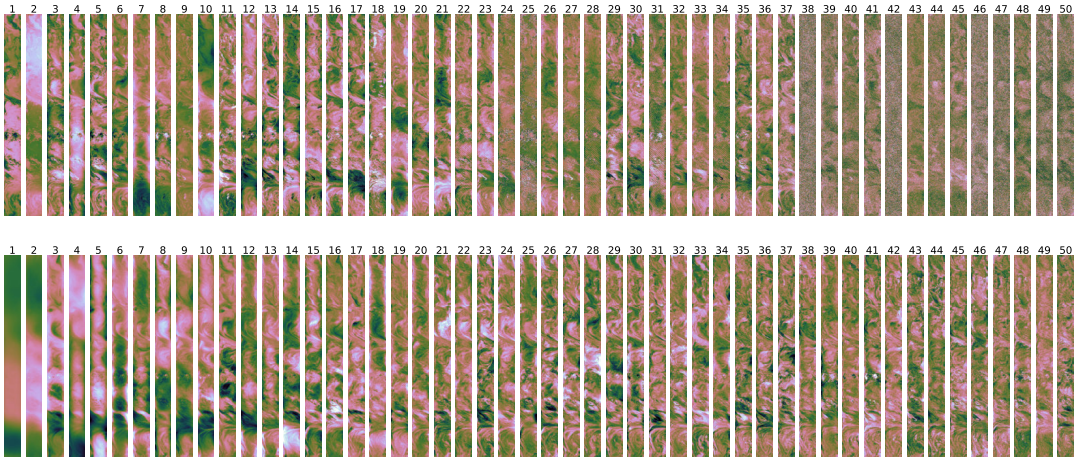


Fig. 3. Each strip is half an orbit projected onto the different components (PCA top, MNF bottom).

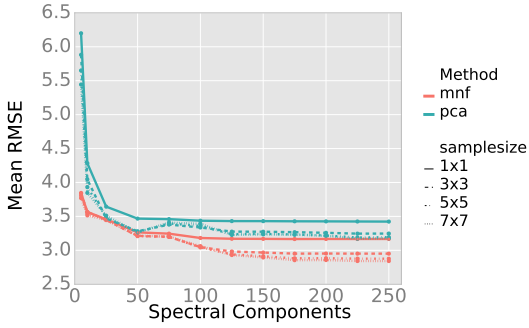


Fig. 4. Mean of RMSE on temperature profiles for different decomposition methods and spatial sample sizes, when including an increasing amount of spectral components. The Y-axis is the mean of the root-mean-square-error on prediction with $o = 90$ i.e. predition of 90 altitudes in the atmosphere given by their pressure level.

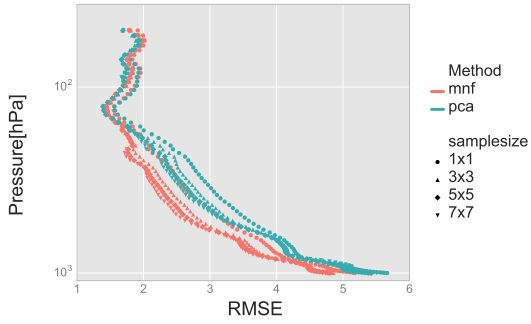


Fig. 5. RMSE on temperature profiles for 90 different altitudes shown by the air pressure (y-axis). In this experiment 175 spectral components was used to fit the Linear Regression model for 3 different neighborhood sampling size on PCA and MNF transformation. Measurements on clouds yields typically higher error rates for lower altitudes. Note that in this experiment nothing has been done to filter away measurements dominated by clouds.

[9] Allan Aasbjerg Nielsen, "An extension to a filter implementation of a local quadratic surface for image noise estimation," in *Image Analysis and Processing, 1999. Proceedings. International Conference on*. IEEE, 1999, pp. 119–124.

[10] M. Studený and J. Vejnarová, *The Multiinformation Function as a Tool for Measuring Stochastic Dependence*, pp. 261–297, Springer Netherlands, Dordrecht, 1998.

[11] V. Laparra, G. Camps-Valls, and J. Malo, "Iterative gaussianization: From ica to random rotations," *IEEE Transactions on Neural Networks*, vol. 22, no. 4, pp. 537–549, April 2011.

[12] Valero Laparra and Raúl Santos-Rodríguez, "Spatial/spectral

information trade-off in hyperspectral images," in *2015 IEEE International Geoscience and Remote Sensing Symposium (IGARSS)*. IEEE, 2015, pp. 1124–1127.

[13] H. Wold, "Non-linear estimation by iterative least squares procedures," *Research papers in Statistics*, pp. 411–444, 1966.

Paper E - Statistical Retrieval of Atmospheric Profiles with Deep Convolutional Neural Networks

Statistical Retrieval of Atmospheric Profiles with Deep Convolutional Neural Networks

David Malmgren-Hansen[a], Valero Laparra[b], Allan Aasbjerg Nielsen[a], Gustau Camps-Valls[b]

[a]*Technical University of Denmark, Lyngby, Denmark*
[b]*Image Processing Laboratory (IPL), Universitat de València, València, Spain*

Abstract

Infrared atmospheric sounders, such as IASI or AIRS, provide an unprecedented source of information for atmosphere monitoring and weather forecasting. The obtained products represent a significant improvement in the accuracy and quality of the measurements used for meteorological models. Sensors provide rich spectral information that allows retrieval of temperature and moisture profiles, ass well as relevant trace gases. From a statistical point of view, the challenge is immense: on the one hand, "underdetermination" (i.e., too many parameters and too few observations) is common place as regression needs to work on high dimensional input and output spaces; on the other hand, redundancy is present in all dimensions (spatial, spectral and temporal). On top of this, large amounts of noise sources are encountered in the data. In the last decade, machine learning has emerged as a proper framework to tackle these data problems, and lately deep learning has excelled in many classification problems. Few developments are found in the field of regression.

In this paper, we present for the first time the use of deep convolutional neural networks for the retrieval of atmospheric profiles from IASI sounding data. The proposed scheme performs multidimensional nonlinear output regression, accounts for noise features, and exploits correlations in all dimensions. The first step of the proposed pipeline performs spectral dimensionality reduction taking into account the signal to noise characteristics. The second step encodes spatial and spectral information, and finally pre-

Email address: dmal@dtu.dk (David Malmgren-Hansen)

diction of multidimensional profiles is done with deep convolutional networks. We give empirical evidence of the performance in a wide range of situations. For this, we collected a big database of co-located IASI radiances and re-analysis temperature profiles. Networks were trained on full orbits and tested out of sample with great accuracy over competing approximations, such as linear spatio-spectral regression (+32%). We also observed a huge improvement in accuracy when predicting over clouds, thus increasing the yield by 34% over linear regression. The proposed scheme is modular and allows us to predict related variables from an already trained model, performing transfer learning in a very easy manner. We conclude that deep learning is an appropriate learning paradigm for statistical retrieval of atmospheric profiles.

Keywords: Deep learning, Machine Learning, Atmospheric parameter retrieval, infrared sounders, IASI, Convnets

---

Contents

Highlights

- Full processing chain proposed: noise-aware dimensionality reduction, spatial-spectral retrieval, and multioutput cross-relations
- Deep convolutional networks excel in atmospheric parameter retrieval
- Computational efficiency and noticeable accuracy gains over clouds
- Networks allow transfer knowledge to predict other state variables easily

1. Introduction

Temperature and water vapour atmospheric profiles are essential meteorological parameters for weather forecasting and atmospheric chemistry studies. Observations from high spectral resolution infrared sounding instruments on board of satellites provide unprecedented accuracy and vertical resolution of temperature and water vapour profiles. However, it is not trivial to retrieve the full information content from radiation measurements. Accordingly, improved retrieval algorithms are desirable to achieve optimal performance for existing and future infrared sounding instrumentation.

The use of MetOp data in Numerical Weather prediction (NWP) accounts for 40% of the impact of all space based observations in NWP forecasts. The Infrared Atmospheric Sounding Interferometer (IASI) sensor is implemented on the MetOp satellite series. Products obtained from IASI data are a significant improvement in the quality of the measurements used for meteorological models. In particular, IASI collects rich spectral information to derive temperature and moisture profiles, which are essential to the understanding of weather and to derive atmospheric forecasts. The sensor provides infrared spectra, from which temperature and humidity profiles with high vertical resolution and accuracy are derived. Additionally, it is used for the determination of trace gases such as ozone, nitrous oxide, carbon dioxide and methane, as well as land and sea surface temperature, emissivity, and cloud properties (EUMETSAT, 2014; Tournier et al., 2002).

EUMETSAT, NOAA, NASA and other operational agencies are continuously developing product processing facilities to obtain L2 atmospheric profile products from infrared hyperspectral radiance instruments, such as IASI, AIRS or the upcoming MTG-IRS. One of the retrieval techniques commonly used in L2 processing is based on linear regression, which is a valuable and very computationally efficient method. It consists of performing ordinary least squares linear regression on top of the data projected onto the first principal components or Empirical Orthogonal Functions (EOF) of the measured brightness temperature spectra (or radiances) and the atmospheric state parameters. To further improve the results of this scheme for retrieval, nonlinear statistical retrieval methods can be applied as an efficient alternative to more costly optimal estimation

(OE) schemes. These methods have proven to be valid in retrieval of temperature, dew point temperature (humidity), and ozone atmospheric profiles when the radiance data are used (Camps-Valls et al., 2012).

From a statistical standpoint, the challenge is immense: on the one hand, "underdertermination" (meaning too many parameters and too few observations) is common place as regression needs to work on high dimensional input and output spaces; on the other hand, redundancy is present in all dimensions (spatial, spectral and temporal). On top of this, several noise sources and high noise levels are encountered in the data, which in many cases are correlated with the signal. The previous L2 processing scheme presented in (Camps-Valls et al., 2012) consisted of first performing a *spectral* dimensionality reduction based on Principal Component Analysis (PCA) (Hotelling, 1933), and then a nonlinear regression based on kernel methods (Camps-Valls et al., 2011a,b; Camps-Valls & Bruzzone, 2009). Despite being an effective approach, the scheme reveals some deficiencies. The PCA transformation accounts for most of the signal variance, but does not consider the correlation between the signal and the noise. On the other hand, the spatial information is discarded and the retrieval algorithm acts on a pixel (FOV) basis. Only very recently methods have included spatial-spectral feature relations in the retrieval algorithm, yet in an indirect way through either post-filtering of the product, or via data compression (García-Sobrino et al., 2017). In this paper, we propose a general scheme to cope with all these problems.

Three main motivations guide our proposal:

- *Accounting for noisy features.* Recently, in (García-Sobrino et al., 2017), great improvement in the performance of retrieval methods was reported when applying standard compression algorithms to the images. Although this result may appear counter-intuitive since compression implies reduction on the amount of information in the images, a certain level of compression is actually beneficial because: 1) compression removes information but also noise, and it could be useful to remove the components with low signal-to-noise ratio (SNR); and 2) spatial compression introduces information about the neighbouring pixels in an indirect yet simple way. The use of Minimum Noise Fractions (MNF) employed

in this paper is a simpler and more mathematically elegant way to take advantage of both properties simultaneously. MNF is specifically designed to sort feature components (loadings) according to the SNR score (Green et al., 1988). In this work we compare the effect of using PCA or MNF when retrieving temperature profiles using IASI data. We show that MNF is better suited to this task. Moreover, since PCA and MNF are both linear and unsupervised transformations, using MNF does not introduce any critical modification in the data processing pipeline. One can simply replace the PCA principal components with (possibly a lower number of) MNF components.

- *Accounting for smoothness in the spatial and vertical dimensions.* All previous algorithms (Blackwell et al., 2008; Camps-Valls et al., 2012, 2016; Laparra et al., 2017) used for statistical retrieval exploited the spectral information in the FOVs only, and discarded spatial information of the acquired scene. Including spatial information in classifiers and regression methods has been done traditionally done via hand-crafted features (Plaza et al., 2002; Tuia et al., 2010; Camps-Valls et al., 2006, 2014). This, however, requires expert knowledge, it is time consuming and scenario dependent. In the last decade, convolutional neural networks (CNNs) has excelled in many classification problems in remote sensing (Aptoula et al., 2016; Geng et al., 2015; Zhang et al., 2016a; Luus et al., 2015; Maggiori et al., 2017; Marmanis et al., 2016; Zhang et al., 2016b; Romero et al., 2016). CNNs allow to easily *learn* the proper filters to process images and optimize a task (in our case, prediction of atmospheric profiles). It is, however, quite striking that very few applications of CNNs are found in the field of regression, and none to our knowledge for bio-geo-physical parameter retrieval. In this paper, we present for the first time the use of deep convolutional neural networks for the retrieval of atmospheric profiles from IASI sounding data. We should note that, neural networks offer an additional advantage to our multivariate regression problem: models are intrinsically multi-output and account for the cross-relations between the state vector at different altitudes. This allows us to attain smoothness, and hence consistency, across the atmospheric column in a

very straightforward way.

- *Accounting for higher level feature representations.* The problem of translating radiances to state parameters is a challenging one because of its intrinsic high nonlinearity and underdertermination. Deep learning offers a simple strategy to approach the problem of complex feature representations by stacking together several convolutional layers. In the last decade, deep networks have taken over shallow architectures in many recognition and detection tasks. This is our third motivation to explore deep convolutional nets in the context of atmospheric parameter retrieval.

Capitalizing on these three motivations, in this paper we propose a chained scheme that exploits the MNF transformation and deep convolutional neural networks for atmospheric parameter retrieval. In summary, the proposed scheme performs multidimensional nonlinear output regression, accounts for noise features, and exploits correlations in all dimensions.

The remainder of the paper is organized as follows. Section 2 presents the processing scheme and analyses the building blocks (dimensionality reduction and retrieval) in detail. Section 2.1 describes the datasets used for the development of the algorithm. Section 3 illustrates the performance of the proposed method in terms of accuracy, bias and smoothness of the estimates (across the space and vertical dimensions), both over land and over ocean. We also pay attention to the yield ratio when predicting over clouds as a function of the cloud fraction. The section ends with an exploratory analysis of the performance of the method to estimate other (yet related) variables with minimal retraining. We outline the conclusions of the work and the foreseeable future developments in Section 4.

2. Methodology

Rather than modeling atmospheric parameters from single point measurements, the purpose here is to investigate spatial dependencies in the retrieval. IASI data are collected as point measurements in a 2×2 grid simultaneously. The IASI instrument

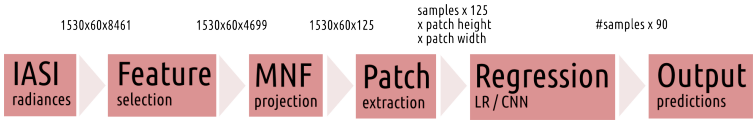


Figure 1: Pipeline schematic: IASI spectra are first reduced from the original 8461 spectral channels by selecting a subset of 4699 channels according to noise specifications in (Camps-Valls et al., 2012), which then pass through an MNF projection to reduce the dimensionality to 125 features. Subsequently, patch extraction is performed with varying sizes. Finally, either a linear regression or a CNN is used for prediction of the atmospheric profiles sampled at 90 vertical positions.

scans a swath of 60 points. This fact can be used to structure the data in rectangular grids and treat them as images likewise (García-Sobrino et al., 2017). We use this approach in two steps of our prediction pipeline illustrated in Figure 1. The pipeline consists of 1) removing irrelevant spectral bands and structuring the data as images of dimension of $1530 \times 60 \times 4699$ per orbit (cf. (Camps-Valls et al., 2012)), 2) applying the linear basis (learned using an MNF decomposition) on the spectral components, 3) extracting patches from data so that observations are local neighbourhoods around each pixel, 4) running either a CNN model or a linear regression for retrieval of atmospheric parameters at 90 different altitudes simultaneously. Let us describe in detail each of these steps.

2.1. Data collection and preprocessing

The Infrared Atmospheric Sounding Interferometer (IASI) is an instrument implemented on the MetOp satellite series. From MetOp's polar orbit, the IASI instrument scans the Earth at an altitude of, approximately, 820 kilometers. The instrument measures in the infrared part of the electromagnetic spectrum (between 645 cm^{-1} and 2760 cm^{-1}) at an horizontal resolution of 12 kilometers over a swath width of, approximately, 2200 kilometers. It obtains a global coverage of the Earth's surface every 12 hours, representing 7 orbits in a sun-synchronous mid-morning orbit. This represents more than one million of high dimensionality samples to be processed each day. Obtaining all the products provided by IASI with classical methods requires an enormous computational load. To obtain these measurements efficiently some works have

focused on using machine learning methods (Camps-Valls et al., 2012; Laparra et al., 2015, 2017).

Each original sample has 8461 spectral bands, but following previous recommendations (Camps-Valls et al., 2012) we performed feature selection removing the most noisy bands and keeping 4699. Even with such drastic feature reduction, regression methods can suffer and easily overfit as many parameters need to be learned. In addition, even though some noise is removed by doing this channel selection, there still remains some noise and spectral redundancy in the data. Actually it has been suggested that simple spatial smoothing techniques remove the noise and help improving the predictions quality (Garcia Sobrino et al., 2017). In the following subsection we pay attention to the feature extraction step to better pose the problem. Each sample is matched with temperature and dew point temperature profiles estimated using the European Centre for Medium-Range Weather Forecasts model. Products obtained from IASI data are a significant improvement in the quality of the measurements used for meteorological models. Profiles of humidity obtained using IASI obtain an error of within 10% and a vertical resolution of one kilometer and profiles on temperature with an accuracy of within one Kelvin.

2.2. Dimensionality Reduction

Traditionally, dimensionality reduction is done by means of PCA, or equivalently by means of Singular Value Decomposition (SVD) (Golub & Van Loan, 1996). In this context, PCA compresses the total variation of the original variables (i.e. radiances) into fewer uncorrelated variates termed 'principal components' which minimize the reconstruction error of the original variables. Alternatively, one could use a different feature extraction method, for instance Independent Component Analysis (ICA) (Hyvärinen et al., 2001) where the not just uncorrelated but statistically independent variates maximize a measure of non-Gaussianity such as negentropy in all original variables. In (Malmgren-Hansen et al., 2017) we recently showed that uncorrelated variates resulting from a minimum noise fraction (MNF) transformation outperform principal components when used as predictors of the atmospheric profile. These MNF variates minimize the noise fraction or equivalently maximize the signal-to-noise ratio (given

a noise model) in all original variables. Here, the noise is estimated as the pixel-wise residual from a quadratic function fitted in a 3x3 window. It can be shown that the MNF variates can be considered as a form of independent components. Figure 2 shows a result from (Malmgren-Hansen et al., 2017) that compares MNF and PCA for analysis at the pixel level (1×1), as well as when local 3×3, 5×5, and 7×7 neighbourhoods are used. It is seen that the performance gain converges above 100 spectral components even for increasing spatial sample sizes in the experiments. We have chosen 125 spectral MNF components for the experiments presented Section 3.

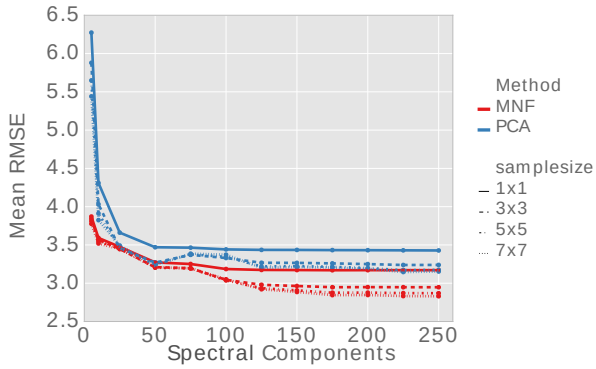


Figure 2: Mean RMSE error for linear regression as a function of number of spectral components included, when predicting atmospheric temperatures. PCA and MNF signal decompositions of spectral bands are compared.

2.3. Regression Models

In this work we use ordinary least squares (OLS) linear regression as a benchmark method to compare the results using CNNs. The linear model, here in its simplest version, is estimating a target variable $\mathbf{t}$ with K elements as,

$$\hat{\mathbf{t}} = \mathbf{f}(\mathbf{x}_n) = \mathbf{W}\mathbf{x}_n + \mathbf{b} \tag{1}$$

where $\mathbf{x}_n$ is the n'th observation of size I input variables and $\mathbf{b}$ the model intercept. In our regression I would equal 125 decomposed spectral radiances times the number

of local neighbourhood pixels (e.g. $125 \times 3 \times 3 = 1125$). Given all N observations, a closed form solution can be found to the minimization of the residuals,

$$\arg\min_{\mathbf{W}} \quad \|\mathbf{t} - (\mathbf{x}_n\mathbf{W} + \mathbf{b})\|^2 \quad for \quad n = 1, .., N \tag{2}$$

This gives a set of independent predictions for all target variables t_k, with $k = (1, ..., K)$. In our regression we are predicting 90 atmospheric temperatures, hence $K = 90$.

If we keep the assumption of output independence and further assume $\mathbf{t}_k$ to be Gaussian distributed and represented by a deterministic function with noise added, $\mathbf{t} = \mathbf{f}(\mathbf{x}_n) + \mathbf{e}_n$, we can see that the likelihood,

$$p(\mathbf{t}|\mathbf{x}_n) = \prod_{k=1}^{K} p(t_k|\mathbf{x}_n), \tag{3}$$

$$p(t_k|\mathbf{x}_n) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left(\frac{(f_k(\mathbf{x}_n) - t_k)^2}{2\sigma^2}\right) \tag{4}$$

reduces to the following error function for maximizing the likelihood over all N observations,

$$E = \sum_{n=1}^{N}\sum_{k=1}^{K}(f_k(\mathbf{x}_n) - t_{k,n})^2 = \|\mathbf{f}(\mathbf{x}_n) - \mathbf{t}_n\|^2 \tag{5}$$

when we take the negative logarithm and remove additive and multiplicative constants, with f_k being a single target of the 90 atmospheric temperatures. This minimizing this error function is as well the most popular approach to regression with neural networks and the difference between our predictor function $\mathbf{f}(\mathbf{x}_n)$, being the linear regression or some neural network denoted $\mathbf{y}(\mathbf{x}_n; \mathbf{W})$ for all layers' weights $\mathbf{W}$, is the complexity. Since $\mathbf{y}(\mathbf{x}_n; \mathbf{W})$ is non-linear, Equation 5 becomes a non-convex problem and there is no longer a closed form solution, (Bishop, 2006). Note that if linearity is kept in the last layer of the neural network, i.e. no non-linear activation function is applied on the output, our model can be written as,

$$\mathbf{y}(\mathbf{x}_n; \mathbf{W}) = \mathbf{W}_L\mathbf{g}(\mathbf{x}_n; \mathbf{W}_{1,..,L-1}) + \mathbf{b}, \tag{6}$$

and we see that the last layer, L, of the neural network is a linear regression on a set of non-linear feature extractions from the previous $L - 1$ layers. When the first layers' weight vectors $\mathbf{w}_{1,..,L-1}$ are given, the last layer weights $\mathbf{w}_L$ can be found with

a closed form solution as with the linear regression. This can be used to ensure the optimal set of parameters for the last layer after CNN training (Bishop, 1995) or used in a hybrid training algorithm as suggested in (Webb & Lowe, 1988).

The error in Equation 5 corresponds to minimizing the variances of our estimated target functions given that each K outputs are independent and can be modelled with one global parameter for the variance,

$$\sigma^2 = \frac{1}{NK} \sum_{n=1}^{N} \sum_{k=1}^{K} (y_k(\mathbf{x}_n; \mathbf{W}) - t_{k,n})^2 = \|\mathbf{f}(\mathbf{x}_n) - \mathbf{t}_n\|^2 \tag{7}$$

This is not necessarily true in our case as one could assume that nearby variables in the vertical atmospheric profile will be correlated. We will keep this assumption in order to ensure simplicity of the objective function, but other approaches could be adopted in future studies.

The purpose of comparing a linear model with a CNN for estimating atmospheric temperatures is to study how the spatial information in the data helps determining the optimal prediction model. In a linear model we can model local input correlations by concatenating a neighbourhood of spectral pixel values when predicting the center pixel. In a CNN all spatial content in the given input patch is mapped to a latent representation through a series of stacked convolutions, for which the kernel coefficients are a part of the parameters we optimize. If, e.g. the proximity of a coastline has a high influence on the target variable for the IASI data, a kernel in the CNN can learn to represent this feature in the latent representation, no matter where in the patch that the coastline appears.

To find the optimal set of weights for the CNN we use an iterative stochastic gradient descent (SGD) based update scheme. It is well known that estimating the error for all training samples in each iteration leads to slower convergence why a mini batch approach is used in the field of deep learning now. This, though, leads to more noisy estimates of the error function and methods to cope with this stochastic noise have been proposed. We use the method called ADAM (Kingma & Ba, 2014), where exponential moving averages of the gradients and squared gradients are used to ensure a smooth convergence of the parameters. Since our initial targeted state vector (temperatures, dew-point temperatures, etc) can have different variances across the atmosphere, one

could chose to normalize the target variables. This might lead to significantly different solutions in an SGD based scheme, as opposed to e.g. an SVD factorization of the ordinary least squares problem. The fact that target values might change scale can have a big impact on some SGD schemes since the gradient term scales as well. Unless accounted for in the learning rate, this will change the convergence of a solution. The ADAM SGD scheme chosen for optimization in our experiments is practically invariant to scaling of the gradients due to its update rule based on first and second order moment vectors. These vectors impose an individual stepsize for each parameter in the network during the iterative parameter updates.

3. Experimental Results

The goal of our experiments is to demonstrate the advantages of CNNs for the retrieval of atmospheric variables from infrared sounders. In particular, we will illustrate how the networks, unlike other machine learning methods, include spatial regularization in a natural way. This feature results in improved prediction in the case of cloud coverage or noisy settings. Another advantage of the method is that cross-relations between the different atmospheric states are captured, so smoothness in the vertical profile is also achieved. Finally, we explore a very interesting possibility of the network to perform transfer learning, by which a network trained for example for temperature profile estimation can be re-used for moisture estimation with minor retraining.

3.1. Experimental setup

We will employ the data collected in 13 consecutive orbits within the same day, 17-08-2013, by the IASI sensor. Each orbit consists of approximately 92,000 samples. We use the first 7 orbits (which cover most of the Earth) for training and the last 6 for testing (which also cover most of the Earth). Figure 3 shows the coverage of the two different sets of data taken on the same day.

3.2. Models

In order to investigate different extents of spatial information, four CNN models have been designed, see Tables 1 and 2. A summary of some essential CNN features

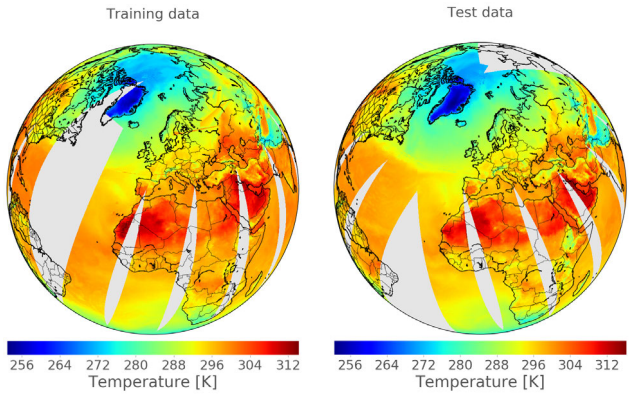


Figure 3: Example of how we split the data: training (left) and test (right). Figures show surface temperatures for different orbits.

are given in Table 1 while a thorough description is found in the Appendix.

Table 1: Table of CNN architectures developed in this work. Training times are reported on a Python (Theano) implementation running on a single NVIDIA Titan Black GPU.

	CNN A	CNN B	CNN C	CNN D
Input Size	125×3×3	125×10×10	125×15×15	125×25×25
Number of layers	4	6	6	7
Number of parameters	127,290	347,070	639,750	938,350
Output dimension	90			
Optimizer	ADAM (Kingma & Ba, 2014)			
Approx. Training time	4h	11h	18h	39h
# train. samples	524,552	460,887	415,472	324,792
Mean test RMSE [K]	2.48	2.43	2.19	2.28

Our experiments consist of comparisons between CNN predictions with an ordinary least squares (OLS) linear regression model, yet including spatial information in the OLS too. The linear model defined according to Equation 1 can be extended to different spatial sample size by appending new variables to the columns of the data matrix $\mathbf{X}$.

As the input dimensionality rapidly grows with increasing p, the size of the dataset sets a natural limit to the spatial extend that can be included in the regression. We have therefore limited the OLS regression to $p = 15$. In particular, CNN A is trained on patches of $p = 3$. With a convolution kernel size, s, in the first layer of 3×3 coefficients this gives one valid convolution per patch. Practically this is equivalent to multilayer perceptron network with the nine neighbourhood pixels stacked in an input vector. In CNN B, C, D we keep s as 3×3 filters, while letting the patch size increase resulting in increased number of convolutions across the patch, i.e. we model local correlations (features) across an entire patch.

3.3. Retrieval performance and evaluation

In Table 2 the mean over the RMSE vertical profiles are given for our regression models with different sizes of p with the corresponding individual profiles shown in Fig. 4. In general, CNNs outperform linear regression models in terms of feature extraction, but this result can be further improved when training with the ADAM scheme is done. According to Equation 6 we can find the last CNN layer weights with the OLS algorithm when we fix the previous layers' weights. In our experiments, this procedure improved the CNN predictions with additionally $12 - 17\%$ for all CNN architectures.

Table 2: Summary of the mean RMSE (across the atmosphere profile) on temperature prediction. The CNN + Opt. row is the same network as the first but where the last layer is optimized with the closed form least square solution after training.

Patch Size	1×1	3×3	5×5	7×7	10×10	15×15	25×25
CNN	–	2.48	–	–	2.43	2.20	2.28
CNN + Opt.	–	2.11	–	–	2.01	1.94	2.01
OLS	3.30	3.00	2.91	2.86	2.84	2.85	–

Let us now analyse some key aspects of the proposed CNN models: 1) the smoothness of the prediction profiles across space and vertical dimensions, and 2) the transferability of the models to be re-used in predicting other variables.

During a neural network optimization the average gradient of the error function is back propagated to update the weights, Equation 5. In this way we capture the best

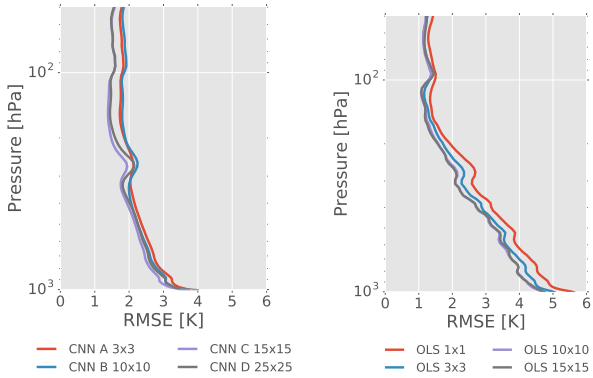


Figure 4: RMSE error profiles of model prediction for both CNNs and linear models at different input spatial patch sizes. The CNN generally outperform OLS regression except at very high altitudes. The temperatures at lower altitudes (> 200 hPa) are the most important for meteorological models.

average solution to our regression. This is in contrast to a linear model where each output is an independent model of the input. In the case of atmospheric parameter retrieval where neighbouring targets are spatially dependent, the average gradient or the non-linearity in the CNN seem to smooth vertical predictions as well. Figure 5 shows 4 transects of the mean error for a given path in an orbit of data. It can be seen that the linear model can obtain spatial smooth (horizontally) predictions by increasing the input patch size. The CNN ensures a smooth error profile both in the vertical and horizontal directions of the transect. The estimated cloud fraction is marked on each pixel of our dataset and can be seen as the white dashed line in Figure 5. Though higher errors are generally expected in cloudy areas it seems that the correlation between the cloud fraction and the error are weak. We shall explore this further.

3.3.1. Predictions over clouds

Predictions are inherently disturbed by strong attenuation or mixing of radiometric contributions from a large number of sources. It is commonly known that the presence of clouds attenuate the signal and hamper retrieval of parameters. Some approaches to temperature prediction typically act on cloud-free marked pixels only to be confident

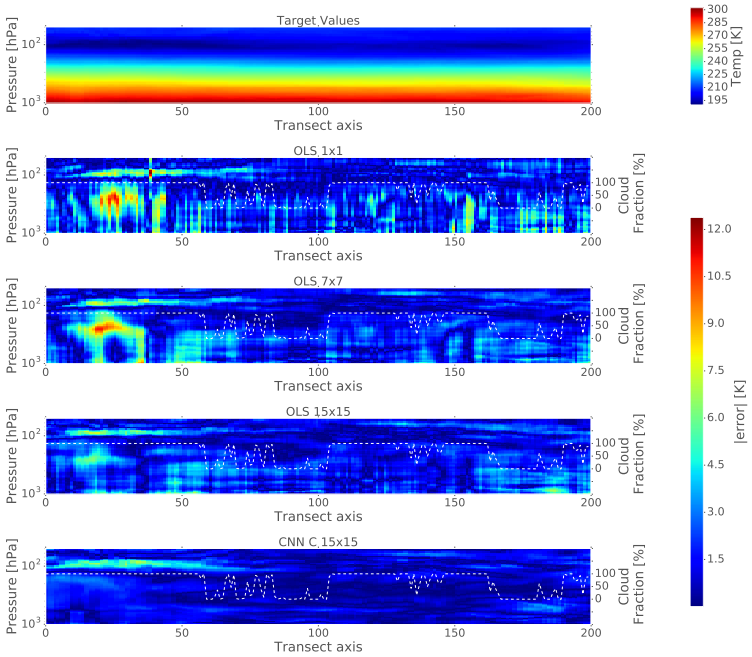


Figure 5: Top plot shows the target temperature along a transect profile, lower four plots shows the transect profile of the prediction error from different regression models. White dashed line is the cloud fraction, i.e. the percentage on cloud each input sample is marked with. The y-axis is the altitude pressure level.

on the obtained predictions. Cloud masks can to some extent be estimated from optical sensors, but different approaches to generating cloud masks can have high influence on the final result[1]. Since we are predicting the center pixel profile from a neighbourhood of pixels one could filter patches based on the amount of clouds. Nevertheless, in the results shown here, no such pre-filtering was performed, but CNNs figure out how to exploit the (possibly less cloudy) neighbouring radiances. Figure 6 shows the error of a CNN on pixels with less than 50% clouds and pixels above. The cloud mask

[1]See e.g. the IAVISA exercise: http://www.brockmann-consult.de/iavisa-info-web/index.html.

contains mostly 0% and 100% cloud fractions. For linear regression the difference between predicting over clouds or in cloud free areas is clear, around an increment of one degree of the error in lower atmospheric layers. In the case of CNNs this difference is less noticeable, around 0.25 degrees in the same area. An important thing to stress is that the CNNs model obtains less prediction error over cloudy areas than the linear model does over cloud free areas.

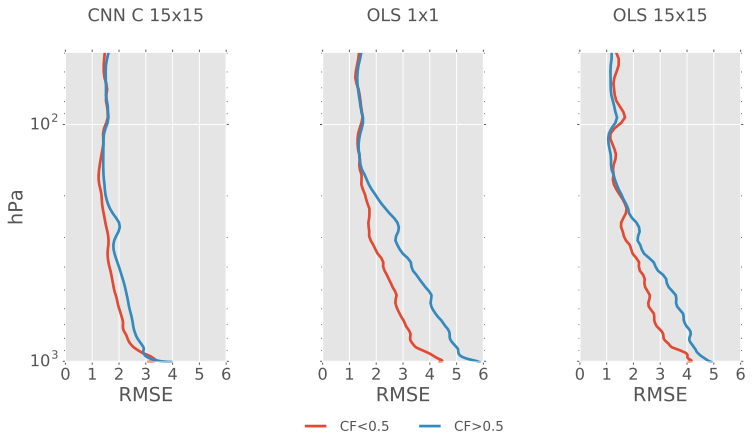


Figure 6: Difference on RMSE profile when testing on cloudy samples (CF>0.5) versus samples marked cloud free (CF<0.5).

3.3.2. Predictions over land

Prediction over land is typically more challenging than over ocean, mainly due to the more varying conditions, landscape and land cover, and changes in bodies' emissivities. We aimed to study the performance of algorithms as a function of the land cover per pixel. Figure 7 shows the error in predictions from a linear model and a CNN with 15×15 pixel input patch size, conditioned on the land fraction. The land fraction mask contains mostly 0% or 100% values, but some coastal areas are given as intermediate values due to the resolution cell covering both land and sea. On the other hand, the land fraction has a high influence on the predictions, and continues to be a challenge

for precise predictions of atmospheric temperature profiles.

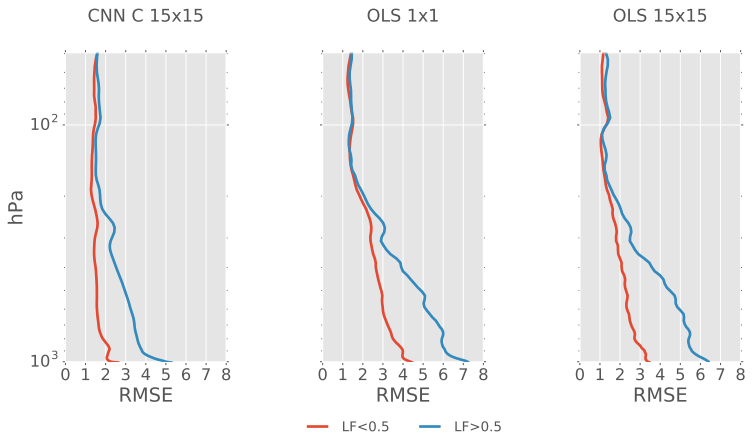


Figure 7: Difference in performance when predicting temperature profiles over land (LF>0.5) and over sea (LF<0.5).

3.4. Transfer Learning

The concept of transfer learning within deep learning has proven useful for a range of computer vision tasks. Deep CNNs trained on large databases of natural images can be transferred to smaller datasets for specific applications with high end performance. There are two overall different approaches to transfer learning. One, as in (Yosinski et al., 2014), where the training of a Network is repeated on a new dataset but starting with the weights found solving the first problem. The second is to consider a part of the network a feature extractor and access the latent representation learned from one dataset and classifier to solve a problem on a second related dataset (Sharif Razavian et al., 2014).

The purpose of exploring transfer learning in our setup is not to unveil whether cross domain features can be learned. Instead, we explore the ability of a model trained to predict atmospheric temperature profiles to be transferred to other output variables, such as moisture profiles. Possible benefits are shorter training time, as well as higher

accuracy.

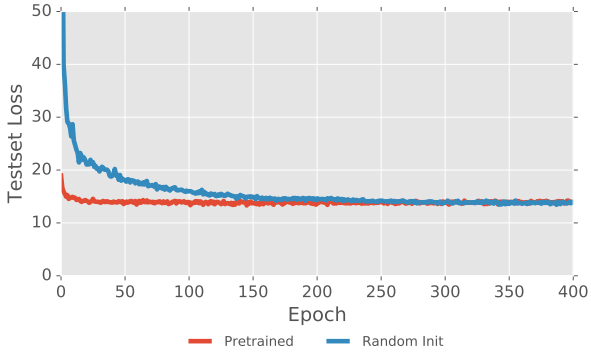


Figure 8: Test error convergence during training for dewpoint temperature prediction. Blue curve is a CNN initialized with random weights and the red is a CNN initialized with the weights for a model that predicts air temperatures. Both models converge to a mean RMSE error of 3.34 K after 400 epochs.

Figure 8 explores speed up of training convergence on predicting dew point temperatures when considering initialization of a CNN with either weights from a network trained on atmospheric temperatures or a standard random initialization for training from scratch. The figure shows that the performance reached by CNN initialized from random weights can be reached in less than around $\frac{1}{8}$ (50 out of 400) of the training time if the weights are transferred from a model trained for another output variable.

Considering a model trained on atmospheric temperatures, a feature extractor for a linear regression to predict dewpoint temperatures can as well be done, and this approach is conceptually closer to the one proposed in (Sharif Razavian et al., 2014). RMSE profiles from the transfer learning experiments are shown in Figure 9.

The red and blue profiles in Figure 9 show that we reach the same performance whether we start with a model trained on atmospheric temperatures or random weights, when predicting dewpoint temperatures. This is not surprising since it is the same dataset we fit the models on, all we change is the predictor variable. When considering the second transfer learning approach where a CNN trained on temperature prediction is used as a feature extractor with a linear regression to predict dewpoint we reach a

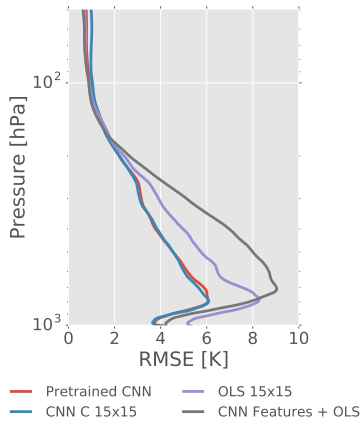


Figure 9: Dew point temperature RMSE profiles of transfer learning regression models. The features learned on temperatures are poor for dewpoint prediction (grey profile) unless fine tuned (red profile).

less optimal solution. The purple profile in Figure 9 shows the second transfer learning approach and training a linear regression directly on the input radiance is shown as the grey profile. At low altitudes we get better accuracy than the shallow linear regression model (>1°RMSE terms). At higher altitudes though, the second transfer learning approach does it worse. Fine tuning for a specific output variable is necessary in order to achieve good predictions.

4. Conclusion

We present for the first time the use of deep convolutional neural networks for the retrieval of atmospheric profiles from infrared sounding data, particularly for IASI data. The proposed scheme performs multidimensional nonlinear output regression, accounts for noise features, and exploits correlations in all dimensions. Good experimental results were obtained over competing approaches in a wide range of situations. Networks were trained on full orbits and tested out of sample with great accuracy. We also observed a huge benefit in accuracy when predicting over clouds, increasing the yield by 34% over linear regression. The proposed scheme is modular and allows us

to predict related variables from an already trained model. We also illustrated this by exploiting the learned network to predict temperature profile and retraining the last linear layer to fit moisture (dew point temperature) profiles. Good results were obtained too, which demonstrates that the learned features by the network impose a sort of spatial and vertical smootheness that can be exploited for other state variables that share these features, such as some trace gases as well. We conclude that deep networks is an appropriate learning paradigm for statistical retrieval of atmospheric profiles.

There are several aspects of the modeling to explore in the future to improve the statistical retrieval. It would be relevant to explore model architectures that directly model the output correlations. This could be done with the neural network by including the joint probabilities between neighbouring targets in expense of a more complicated error function. Alternatively one can predict the difference between neighbouring target variables rather that their value and, in this way, incorporate neighbourhood correlation in the targets. We have shown that there is a high potential for models that incorporate feature extracting abilities as well as capabilities of modeling non-linear phenomena in statistical retrieval. Finding optimal architectures for CNNs remains an open task in the deep learning literature, and due to the non-convexity of the problem, experiments are the only way to find optimal models. In this work, a few architectures have been explored but a larger analysis of this is highly relevant for the application. Further, results in this work were limited by the dataset size and constructing larger datasets that capture more variances, such as (monthly, yearly) temporal variations is needed regardless of the chosen method. Recent alternatives on efficient training of convolutional nets could resolve the induced complexity (Giusti et al., 2013; Sermanet et al., 2013; Kampffmeyer et al., 2016).

ACKNOWLEDGEMENTS

This work was supported in part by the Spanish Ministry of Economy and Competitiveness and by the European Regional Development Fund under Project TIN2015-71126-R, and by the European Research Council (ERC) under Consolidator Grant SEDAL ERC-2014-CoG 647423.

We want to thank Dr. Thomas August (EUMETSAT, Germany) and Xavier Calbet (AEMET, Spain) for the provided data and the fruitful discussions on atmospheric retrievals.

References

Aptoula, E., Ozdemir, M. C., & Yanikoglu, B. (2016). Deep learning with attribute profiles for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, *13*, 1970–1974.

Bishop, C. M. (1995). *Neural networks for pattern recognition*. Oxford university press.

Bishop, C. M. (2006). Pattern recognition. *Machine Learning*, *128*.

Blackwell, W., Pieper, M., & Jairam, L. (2008). Neural network estimation of atmospheric profiles using AIRS/IASI/AMSU data in the presence of clouds. In A. M. L. M. J. L. M. Suzuki (Ed.), *Multispectral, Hyperspectral, and Ultraspectral Remote Sensing Technology, Techniques, and Applications II, Proceedings of SPIE Vol. 7149*. Bellingham, WA.

Camps-Valls, G., & Bruzzone, L. (Eds.) (2009). *Kernel methods for Remote Sensing Data Analysis*. UK: Wiley & Sons.

Camps-Valls, G., Gomez-Chova, L., Munoz-Mari, J., Vila-Frances, J., & Calpe-Maravilla, J. (2006). Composite kernels for hyperspectral image classification. *IEEE Geoscience and Remote Sensing Letters*, *3*, 93–97. Cited By 341.

Camps-Valls, G., Muñoz Marí, J., Gómez-Chova, L., Guanter, L., & Calbet, X. (2011a). Nonlinear statistical retrieval of atmospheric profiles from MetOp-IASI and MTG-IRS infrared sounding data. *IEEE Trans. Geosc. Rem. Sens.*, *49*.

Camps-Valls, G., Munoz-Mari, J., Gomez-Chova, L., Guanter, L., & Calbet, X. (2012). Nonlinear statistical retrieval of atmospheric profiles from MetOp-IASI and MTG-IRS infrared sounding data. *IEEE Transactions on Geoscience and Remote Sensing*, *50*, 1759–1769.

Camps-Valls, G., Tuia, D., Bruzzone, L., & Benediktsson, J. (2014). Advances in hyperspectral image classification: Earth monitoring with statistical learning methods. *IEEE Signal Processing Magazine*, *31*, 45–54.

Camps-Valls, G., Tuia, D., Gómez-Chova, L., Jiménez, S., & Malo, J. (2011b). *Remote Sensing Image Processing. Synthesis Lectures on Image, Video, and Multimedia Processing*. Morgan & Claypool Publishers.

Camps-Valls, G., Verrelst, J., & Muñoz-Marí (2016). A survey on gaussian processes for earth observation data analysis: A comprehensive investigation. *IEEE Geoscience and Remote Sensing Magazine*, .

EUMETSAT (2014). *IASI Level 1: Product Guide*. EUM/OPS-EPS/MAN/04/0032,.

Garcia Sobrino, J., Serra-Sagrista, Laparra, V., Calbet, X., & Camps-Valls, G. (2017). Statistical atmospheric parameter retrieval largely benefits from spatial-spectral image compression. *IEEE Transactions on Geoscience and Remote Sensing*, *PP*, 1–12.

García-Sobrino, J., Serra-Sagristà, J., Laparra, V., Calbet, X., & Camps-Valls, G. (2017). Statistical atmospheric parameter retrieval largely benefits from spatial–spectral image compression. *IEEE Transactions on Geoscience and Remote Sensing*, *55*, 2213–2224.

Geng, J., Fan, J., Wang, H., Ma, X., Li, B., & Chen, F. (2015). High-resolution sar image classification via deep convolutional autoencoders. *IEEE Geoscience and Remote Sensing Letters*, *12*, 2351–2355.

Giusti, A., Ciresan, D. C., Masci, J., Gambardella, L. M., & Schmidhuber, J. (2013). Fast image scanning with deep max-pooling convolutional neural networks. In *Image Processing (ICIP), 2013 20th IEEE International Conference on* (pp. 4034–4038). IEEE.

Golub, G. H., & Van Loan, C. F. (1996). *Matrix computations* volume 3. JHU Press.

Green, A. A., Berman, M., Switzer, P., & Craig, M. D. (1988). A transformation for ordering multispectral data in terms of image quality with implications for noise removal. *IEEE Transactions on geoscience and remote sensing*, *26*, 65–74.

Hotelling, H. (1933). Analysis of a complex of statistical variables into principal components. *Journal of educational psychology*, *24*, 417.

Hyvärinen, A., Karhunen, J., & Oja, E. (2001). *Independent component analysis*. John Wiley & Sons.

Kampffmeyer, M., Salberg, A.-B., & Jenssen, R. (2016). Semantic segmentation of small objects and modeling of uncertainty in urban remote sensing images using deep convolutional neural networks. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 1–9).

Kingma, D., & Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, .

Laparra, V., Malo, J., & Camps-Valls, G. (2015). Dimensionality reduction via regression in hyperspectral imagery. *IEEE Journal of Selected Topics in Signal Processing*, *9*, 1026–1036.

Laparra, V., Muñoz-Marí, J., Gómez-Chova, L., Calbet, X., & Camps-Valls, G. (2017). Nonlinear statistical retrieval of surface emissivity from iasi data. In *IEEE International and Remote Sensing Symposium (IGARSS)*.

Luus, F. P. S., Salmon, B. P., van den Bergh, F., & Maharaj, B. T. J. (2015). Multiview deep learning for land-use classification. *IEEE Geoscience and Remote Sensing Letters*, *12*, 2448–2452.

Maggiori, E., Tarabalka, Y., Charpiat, G., & Alliez, P. (2017). Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, *55*, 645–657.

Malmgren-Hansen, D., Laparra, V., Aasbjerg Nielsen, A., & Camps-Valls, G. (2017). Spatial noise-aware temperature retrieval from infrared sounder data. *IEEE International Geoscience and Remote Sensing Symposium*, .

Marmanis, D., Datcu, M., Esch, T., & Stilla, U. (2016). Deep learning earth observation classification using imagenet pretrained networks. *IEEE Geoscience and Remote Sensing Letters*, *13*, 105–109.

Plaza, A., Martínez, P., Pérez, R., & Plaza, J. (2002). Spatial/spectral endmember extraction by multidimensional morphological operations. *IEEE Transactions on Geoscience and Remote Sensing*, *40*, 2025–2041.

Romero, A., Gatta, C., & Camps-Valls, G. (2016). Unsupervised deep feature extraction for remote sensing image classification. *Geoscience and Remote Sensing, IEEE Transactions on*, *54*, 1349–1362.

Sermanet, P., Eigen, D., Zhang, X., Mathieu, M., Fergus, R., & LeCun, Y. (2013). Overfeat: Integrated recognition, localization and detection using convolutional networks. *arXiv preprint arXiv:1312.6229*, .

Sharif Razavian, A., Azizpour, H., Sullivan, J., & Carlsson, S. (2014). Cnn features off-the-shelf: an astounding baseline for recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops* (pp. 806–813).

Tournier, B., Blumstein, D., Cayla, F., , & Chalon, G. (2002). IASI level 0 and 1 processing algorithms description. In *Proc. of ISTCXII Conference*.

Tuia, D., Ratle, F., Pozdnoukhov, A., & Camps-Valls, G. (2010). Multisource composite kernels for urban-image classification. *IEEE Geoscience and Remote Sensing Letters*, *7*, 88–92. Cited By 42.

Webb, A., & Lowe, D. (1988). *A hybrid optimisation strategy for adaptive feed-forward layered networks*. Technical Report DTIC Document.

Yosinski, J., Clune, J., Bengio, Y., & Lipson, H. (2014). How transferable are features in deep neural networks? In *Advances in neural information processing systems* (pp. 3320–3328).

Zhang, F., Du, B., & Zhang, L. (2016a). Scene classification via a gradient boosting random convolutional network framework. *IEEE Transactions on Geoscience and Remote Sensing*, *54*, 1793–1802.

Zhang, L., Zhang, L., & Du, B. (2016b). Deep learning for remote sensing data: A technical tutorial on the state of the art. *IEEE Geoscience and Remote Sensing Magazine*, *4*, 22–40.

APPENDIX

Table 3: Table of CNN architectures. B.R.D. is a concatenation of 3 layers, Batch Normalization, Rectified Linear Unit activation layer and Dropout. Dropout is performed with a probability $p = 0.5$ in all cases

CNN A			CNN B		
Type	Parameters	Output	Type	Parameters	Output
Input	-	125x3x3	Input	-	125x10x10
Conv	125x60x3x3	60x1x1	Conv	125x60x3x3	60x10x10
B.R.D.	-	60x1x1	Conv	60x60x3x3	60x10x10
Conv	60x120x1x1	120x1x1	Pool	-	60x5x5
B.R.D.	-	120x1x1	B.R.D.	-	60x5x5
Conv	120x240x1x1	240x1x1	Conv	60x120x3x3	120x5x5
B.R.D.	-	240x1x1	Conv	120x120x3x3	120x3x3
Conv	240x90x1x1	90x1x1	Pool	-	120x1x1
			B.R.D.	-	120x1x1
			Conv	120x240x1x1	240x1x1
			B.R.D.	-	240x1x1
			Conv	240x90x1x1	90x1x1

CNN C			CNN D		
Type	Parameters	Output	Type	Parameters	Output
Input	-	125x15x15	Input	-	125x25x25
Conv	125x100x3x3	100x15x15	Conv	125x100x3x3	100x23x23
Conv	100x100x3x3	100x13x13	Conv	100x100x3x3	100x21x21
Pool	-	100x6x6	Pool	-	100x10x10
B.R.D.	-	100x6x6	B.R.D.	-	100x10x10
Conv	100x160x3x3	160x4x4	Conv	100x160x3x3	160x8x8
Conv	160x160x3x3	160x2x2	Conv	160x160x3x3	160x6x6
Pool	-	160x1x1	Pool	-	160x3x3
B.R.D.	-	160x1x1	B.R.D.	-	160x3x3
Conv	160x240x1x1	240x1x1	Conv	160x200x3x3	200x1x1
B.R.D.	-	240x1x1	B.R.D.	-	200x1x1
Conv	240x90x1x1	90x1x1	Conv	200x240x1x1	240x1x1
			B.R.D.	-	240x1x1
			Conv	240x90x1x1	90x1x1

Paper F - Compressed Feature Visualizations in Convolutional Neural Networks

Noname manuscript No.
(will be inserted by the editor)

Compressed Feature Visualizations in Convolutional Neural Networks

David Malmgren-Hansen · Allan Aasbjerg Nielsen · Rasmus Engholm

Received: date / Accepted: date

Abstract Convolutional Neural Networks (Convnets) have achieved good results in a range of computer vision tasks in recent years. Their ability to learn features from large quantities of data provides a strong framework, but it remains a challenging task to interpret these features. Current methods of interpreting Convnets are based on visualizing each node of the network, but modern Convnets can have thousands of nodes. We propose a generic visualization framework based on clustering internal representations across layers with Dirichlet Process Gaussian Mixture Models. Our method compresses the high number of nodes in a Convnet and provides a single cluster result per layer of the Convnet. With this method, one is able in a layer-wise manner to visualize how the network structures information from images. We show in this paper how it can explain the high adaptability of Convnets trained on large scale images databases when transferred to other image problems in different domains, i.e. Transfer Learning. Our results reveal that a large part of a network has structured feature representation of the data at hand despite the fact that the data is very different from the data the Convnet was trained on.

Keywords: Deep learning; Feature visualization; Dirichlet Processes; Gaussian Mixture Models;

David Malmgren-Hansen
Technical University of Denmark
Department of Applied Mathematics and Computer Science
E-mail: dmal@dtu.dk

Allan Aasbjerg Nielsen
Technical University of Denmark
Department of Applied Mathematics and Computer Science
E-mail: alan@dtu.dk

Rasmus Engholm
Terma A/S
E-mail: rae@terma.com

1 Introduction

Convolutional Neural Networks (Convnets) have had a great impact on a range of computer vision problems such as image classification, object detection, image captioning etc. The ability to represent image context by representing it with hierarchically ordered feature extractions makes Convnets suitable for scaling to large complex problems with many categories of data. The reason for this is that early layer features will be simple building blocks in the representation of many types of data, while deeper layer features are more specific to a certain context which make it possible to separate images by predefined classes.
There are rarely any built-in constraints on the structure or location of internal representations in Neural Networks. The idea is to train a Convnet to be a "good" hierarchically ordered feature extractor from large quantities of data. Whether this actually happens and which internal nodes or layers become certain types of feature extractors are very important to explain a model's ability to generalize. It can also give insight into how features learned on one dataset can be transferred to other problems, known as Transfer Learning. Transfer Learning has been heavily explored within deep learning recently and has solved several tasks well, e.g. in [20], [11], [23], [16], [21].

1.1 Related Work

Plotting the values of feature maps in a Convnet (square boxes in Figure 1), such as in [10] is one way of gaining insight, but it leaves the user to interpret a large number of abstract feature maps. Convnets can easily have several hundreds of feature maps in just one layer of the network. In [9] it was shown that the first layer filter parameters resembled Gabor wavelet kernels. This was interesting in the sense that large amounts of image data and a classification task with 1000 classes had forced the filters to become, what has long been known to be good image texture descriptors, [4], [13]. However, showing filter parameters or layer outputs as images becomes harder after the first layer of a Convnet. In deeper layers each node has a number of filters corresponding to the number of channels (feature maps) of the previous layer and are therefore no longer a color specific representation as in the first layer, see illustration in Figure 1. This has led to other recently proposed techniques such as in [23], [22], [12]. In [23] a method is proposed where an internal representation from a node in a Convnet is projected back into input RGB-space by inverting all operations. The technique shows easy interpretable visualizations of a node, however the network used had more than 9000 nodes. If one has a labelled dataset, a way to select interesting nodes could be to find the ones that have high activations values when a group in the dataset, e.g. cats or dog, are processed. Without labelled images though, it becomes a tedious task to analyze all nodes in the network.
The contributions of this paper is twofold. First, we present a framework for visualizing internal representations from clustered feature maps. The method copes with the challenges of dealing with the high number of nodes in a Convnet. Further, it clusters in an unsupervised manner without specifying the number of clusters, and in this way is very flexible across different visualizations purposes. In the second part of our contribution we apply our method on different datasets and show how it can explain the strong ability to generalize that Convnets trained on large scale image recognition problems have. Recent research in transfer learning has shown that these networks generalize to other datasets though their are very different from the dataset the Convnet was trained on, but intuition on why this is has until now not been given.

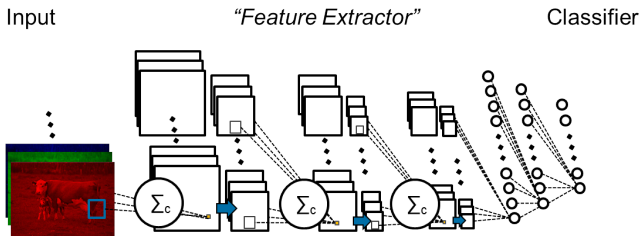


Fig. 1: Example illustration of a Convnet (note the illustration above follows a different architecture than the one used in our experiments). Summation signs over c is sum over c input channels. Squares denote feature maps, i.e. outputs from convolutional layers. The blue arrows represent the chosen subsampling scheme e.g. max-pooling, and the small circles are neurons in fully connected layers. Connections are only shown for the first node in each layer, but all nodes in all layers are connected to all inputs from previous layers.

In [20], different training procedures in Transfer Learning are experimentally explored. The authors randomly split a dataset in two parts and considered transfer learning of a Convnet from one part to the other. In this procedure it is clear that a high amount of learning should be transferable since data originates from the same set. However, it has also been shown that Transfer Learning can improve performance between datasets that are very dissimilar [16], [14]. The question is whether it is only the very basic feature extractions, i.e. early layers, that improve performance when using Transfer Learning with these more dissimilar sets. We will explore this with a new visualization technique in our experiments.

1.2 Proposed Method

In this paper we propose an alternative approach to Convnet visualization based on Dirichlet Process Gaussian Mixture Model (DPGMM) clustering. The DPGMM follows the method in [3], and we suggest to cluster feature representations across nodes in each layer. Different clustering algorithms might be considered, however they need to find the number of clusters automatically. Since we cannot assume anything about the true number of clusters in the feature space of a given Convnet layer, classical algorithms such as K-means and standard Gaussian Mixture Models are not feasible. One advantage of the DPGMM is that points are modelled as belonging to underlying probability distributions with individual covariance matrices. This is in contrast to simpler approaches where points are separated with distance metrics. The Density Based Spatial Clustering in Applications with Noise algorithm (DBSCAN), [6], was tried as well. DBSCAN resulted typically in only one or two clusters for a given layer, and is therefore not optimal for this visualization pipeline.

Our visualization approach can be summarized by the following steps,

1. Given a trained Convnet, process one or more images with the Convnet and save all activations.
2. For each layer, structure the activations as multi-dimensional points with each node being an element in the point vector.
3. Run the clustering scheme over the points and assign each point a label.

4. For feature map activations the labels can be restructured to the feature map size and visualized by discrete a color scheme.
5. For fully connected layers, a vector per image is obtained and the clustering of a set of image vectors reveals the degree of distinguishability from a given layer.

More elaborate explanations of step 3 and 4 are provided in Section 3. For convolutional layers that output feature maps with spatial structure, we consider each feature map pixel as a point and one processed image leads to many points that can be clustered and analyzed. Naturally this approach is infeasible in fully connected layers where each node outputs one value per input image, illustrated as small circles on Figure 1. This gives the practical difference that we can plot the resulting clusters per image as a discretely colored label map for convolutional layers as opposed to fully connected layers where we interpret based on how the layer cluster a set of images. Compared to other visualization techniques, our algorithm visualizes data representations from a whole layer at once rather than a single node in the network. This has the advantage that the we do not need to search through thousands of nodes in order to find strong feature detectors for a given image.

The experiments in Section 3 aim to show how features from Convnets trained on a dataset are transferred to new unseen data. We use the proposed algorithm to analyze VGGNet, [17], which is trained on the ImageNet dataset, [5]. VGGNet has 13,416 nodes distributed on 16 layers. Ideally each node represents a feature necessary for solving the classification task posed in ImageNet, however when used with unseen data we cannot know which nodes are relevant. In Section 3 we will present the visualization results on two unseen datasets with different degrees of similarity to ImageNet.

2 Dirichlet Process Gaussian Mixtures Models

The Dirichlet Process Gaussian Mixture Model is a variant of the variational inference scheme for Dirichlet Process Mixtures introduced in [3]. As opposed to a Gaussian Mixture Model, which aims to model observations as coming from a finite number of Gaussian distributions, the DPGMM considers an infinite number of distributions. The Dirichlet Process (DP), in Equation 1, infers the prior of each of the infinite number of distributions. If we consider a random measure G drawn from an DP with an underlying base distribution G_0 given a positive scaling parameter α,

$$\begin{aligned} G|\{\alpha, G_0\} &\sim DP(\alpha, G_0) \\ v_i &\sim G, \quad i \in \{1, ..., N\} \end{aligned} \tag{1}$$

where v_i is the $i'th$ independent random samples of N samples from our DP. When clustering an unknown number of clusters one wish to sample priors from an infinite series of discrete values that sums to 1 and has a decreasing value as the sampling progresses. This is achieved by following a *stick breaking process* as explained in [15]. The process is to break off fractions of the residual on a unit length *stick* as samples. When the number of samples goes towards infinity the residual goes toward zero and the DP prior will thereby infer a limit to the number of clusters found in the data.

$$\begin{aligned} \pi_i(\mathbf{v}) &= v_i \prod_{j=1}^{i-1}(1 - v_j) \\ G &= \sum_{i=1}^{\infty} \pi_i(\mathbf{v}) \delta_{\eta_i} \end{aligned} \tag{2}$$

δ_{η_i} is an indicator function on the mixing components denoted η_i which in our case being represented by a Gaussian distribution in our case. $\mathbf{v}$ is a vector of independent draws from a Beta distribution $\mathbf{v} = \{v_1, v_2, ...\}$, $v_i \sim Beta(1, \alpha)$. The DPGMM has one parameter, α, sometimes referred to as the concentration parameter. This alters the Beta distribution resulting in a change of how fast our component priors will go towards zero. In practice we cannot fit Mixture Models over an infinite number of components but a finite approximation can be made by leaving the last sample taking the rest of the *stick*. The finite approximation introduces another hyper parameter, the maximum number of components. However, this parameter has little influence on the result as long as it is set higher than the number of components found during optimization. If one can increase this parameter and find results with a higher number of mixtures, it was set too low initially.

For our experiments we have used the implementation of the DPGMM algorithm from the Scikit-Learn package for Python. It uses the Expectation-Maximation scheme to assign data points to the different mixtures components iteratively as the posterior probability is maximized.

3 Experiment

In Deep Learning it has become popular to store input data and model parameters in N-dimensional arrays referred to as tensors, [2], [1]. This is especially convenient with Convnets working on images, as your data structure has four dimensions, i.e. number of images, number of channels (e.g. 3 for RGB), image height and image width. A convolutional layer output will be, for each image, a number of feature maps equal to the number of nodes chosen for the layer, which each has a new number of rows and columns. In our experiment we analyze the pretrained Convnet VGGNet from [17] (model D), which classifies fixed size 224x224 RGB input images among 1000 predefined classes. The convolutional layers of this model produce feature maps with number of rows and columns equal to the number of rows and columns in the input, due to their choice of the convolutional layer hyper parameters "stride" and "border mode". "stride", being the pixel offset of the kernel in the convolution process, reduces the output size if set higher than one. "border mode" refers to handling convolutions near the edges. One can e.g. zero pad the input to obtain the same output size as the input. VGGNet has 16 layers and the feature maps' sizes are reduced trough the network with a max-pooling subsampling function between some of the layers. As the first layer in the model has 64 features maps the output size will be $(n, 64, 224, 224)$, with n being the number of images we process with the network. The last three layers are fully connected layers and their output size $(n, K, 1, 1)$ with K being the number of nodes in each of them. To cluster internal representations we need to consider them as points of features. As we are interested in learning what each layer represents, we suggest to cluster across nodes of each layer. For the fully connected layer this means each vector representation of an image is a point. For the convolutional layers' output (feature maps) we need to consider each pixel in them across all feature maps in the layer as a point. As an example, we can consider the second layer in the model, which outputs an array of size $(n, 64, 224, 224)$. We will convert this into a matrix of size $(n \cdot 224^2, 64)$ considering each row a point in feature space. We then cluster the points with the DPGMM and reconstruct the image of labels given from the assigned mixture components. This allows us to visualize a feature image where each color represents a certain cluster of features in the given layer, without knowing the number of different feature clusters that are present.

The values of the feature map pixels and fully connected activations are ranging in an arbitrary interval given by the learned parameters of the model. We scaled the values to range between 0 and 10 to fit the random initialization scheme of DPGMM components. To reduce the computationally load when clustering the high dimensional points, we use a diagonal covariance matrix scheme for the underlying gaussian mixture components.

The VGGNet handles images of different sizes by resizing to a fixed input size of 224x224 pixels. We followed this approach and did bilinear interpolation so the smallest dimension was 224 pixels, and then cropped the center piece along the other dimension. In this way we avoided uneven resizing, but at the cost of sometimes leaving out information along the edges. Another procedure often used in the ImageNet entries is to do multiple crops and average results, but this yields a higher amount of data that needs to be processed.

3.1 Datasets

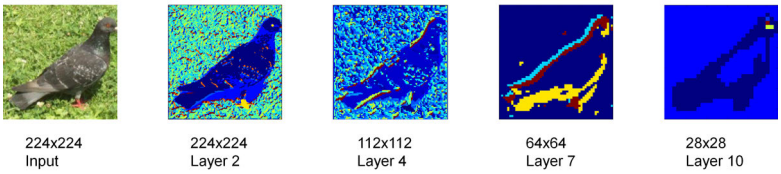
In our experiments we evaluate the presented visualization technique on two datasets, Caltech101 [7], [8] and Warwick-QU [18], [19]. As explained in Section 4, we are visualizing the features learned in the pretrained Convnet VGGnet from [17] that is trained on the ImageNet ([5]) dataset. The two datasets chosen for this experiment are dissimilar to ImageNet in different degrees. Warwick-QU, which consist of Haematoxylin and Eosin (H&E) stained slides of gland samples, is very dissimilar to the natural images of different objects in ImageNet. Caltech101 contains 101 categories of objects which mostly consists of natural RGB images but also includes drawings and grayscale images. This dataset has an overlap with ImageNet objects, e.g. they both contain natural images of pigeons, and we therefore consider it a more similar dataset. Our goal is to explore the extent of Transfer Learning, in the sense of exploring features learned from one dataset on other datasets with different degrees of dissimilarity.

4 Results

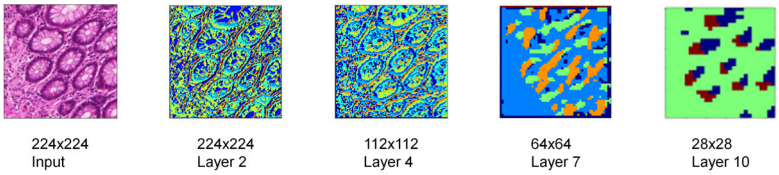
In convolutional layers it is possible to cluster representations from a single image or multiple images, whereas in fully connected layers we have to cluster across several images. In this section we will show results from both approaches on the datasets presented.

In Figure 2, an example is shown with feature map clustering from early to deeper convolutional layers. An example from each dataset is shown, and it can be seen that lower level features get pooled together representing bigger structures deeper in the network. The pigeon sample in the top row of Figure 2 gets more distinctively separated from the background as we go deeper in the network in the sense that the pigeon is represented by fewer clusters. This makes sense, since the background should not be relevant for the object class. The same happens with the cell sample from Warwick-QU and this is despite of the context in the cell image being very far from the context represented in the ImageNet setup. No retraining on the new datasets was performed, so the feature detectors are the ones optimized on ImageNet.

We ran the algorithm on a number of cell images from Warwick-QU and plotted the centers of non background components in their input space position, shown on Figure 3. The point on Figure 3 is not to present a good cell detector, but it is an interesting fact that it naturally arises from accumulated information in layer 10 of the pretrained Convnet. This



(a) Sample from Caltech101



(b) Sample from Warwick-QU

Fig. 2: DPGMM clusters of feature maps from a single image for different layers in the network, where each color is a label given from the clustering. Note that the feature map size decreases due to the Max-pooling subsampling operations. Top row is a Caltech101 sample and the bottom row is a Warwick-QU. In these experiments $\alpha = 0.2$. It should be noted that the colors between layers in above figure have no correspondence since they are assigned in numerical order from the clustering algorithm which changes due to the random mixture initialization.

shows that there is structured context representation all the way up to layer 10 in this model on a new different dataset, since the model can separate cells from the background.

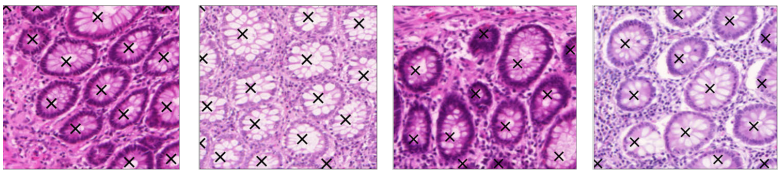


Fig. 3: Non-background component centers from layer 10 plotted in input space for different cell images.

When analyzing the representations in even deeper layers, there is little spatial information about them left in the representation. It is therefore necessary to look at clusters over a range of image examples. From layer 14 in our model, which is a fully connected layer, we have clustered all 165 images in Warwick-QU. The algorithm clusters them as seen in Figure 4.

The Warwick-QU dataset is annotated with grades (benign, malign) and patient ID. Cluster "1" in Figure 4 is 74.4% benign image samples, whereas cluster "2" is 92.9% malign samples. Clusters "3", "4" and "5" are single sample clusters, and are outliers in our cluster-

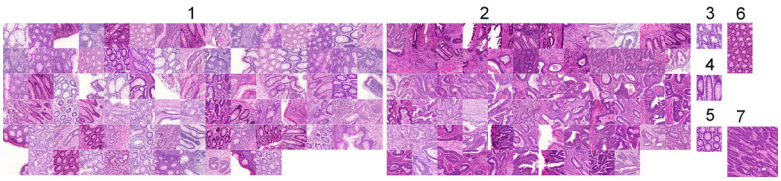


Fig. 4: Clustering over all images of the vector representation from layer 14. In this experiment $\alpha = 0.1$ for the DPGMM clustering.

ing result, they are all benign samples. Cluster ”6” has two benign samples from the same patient and cluster ”7” has four malign samples from another patient. We ran experiments with the concentration parameter α equal to 0.1 and 0.2. With α equal to 0.2, cluster ”1” became 100% benign of size 27 samples, at the cost of cluster ”2” only being 64% malign. In a classification context we do not achieve a high performance, but the model was also never trained for this task. The interesting fact is that training a model on ImageNet gives some ability to discriminate between benign and malign samples in this dataset. What our experiment also shows is that it is not only lower level layers that have potential to contribute to performance gain when using Transfer Learning even between very dissimilar datasets. The feature vector used to cluster the images in Figure 3 was extracted from layer 14 out of 16 layers. This suggests that a large part of the network generalizes well across different types of data.

The original task of the Warwick-QU dataset was to segment the images. It is not in the scope of this paper to explore the process of using a Convnet model trained for classification on a segmentation task, but some approaches on how to do this are presented in [11]. Given our analysis, it seems promising to use a pretrained model for the segmentation task even though the original dataset is significantly different from the Warwick-QU data.

The Caltech101 dataset has more similarity with ImageNet than Warwick-QU. However, it has 101 categories compared to 1000 in ImageNet, and further it contains drawings of objects which is not the case for ImageNet. Clustering of Caltech101 images based on activations from layer 14 with α values 0.2, 1, 1000 yielded 31, 40, 44 clusters, showing that the number of clusters is not very sensitive to this parameter. Due to the stick breaking scheme, where we iteratively sample our prior as a percentage of the residual on a unit length stick combined with the α parameters influence on the beta distribution, the sensitivity on the number of clusters low. Most of the clusters from the experiments with $\alpha = 1$ are shown in Table 1 together with the percentage of Caltech101 categories they contain.

Besides the shown clusters, one cluster contains samples from 82 categories and seems to capture a range of samples from different Caltech101 categories. The clusters in Table 1 contain mainly one Caltech101 category or a combination of categories that are similar, like ”Faces” and ”Faces easy”, ”ketch” and ”schooner”, as well as ”leopards” and ”wild cat”. Figure 5 shows 4 images from some sample clusters. As supplementary material (Online Resource 1), a document with a larger amount of clustered image samples is provided for all clusters.

Simply using the 1000 predefined categories from the original training task of the Convnet to label Caltech101 images does not provide any meaningful grouping of them. We found 697 different labels on Caltech101 by classifying them with ImageNet labels’, and no groups had high correlation with Caltech101 categories. Since ImageNet has a 1000 classes,

Table 1: Table of 39/40 DPGMM cluster results when clustering over all Caltech101 images from layer 14 representations. Left column: percentages of all samples in Caltech101 category. Right column: corresponding Caltech101 category. For example, the first category consists solely of watch pictures and it is 76% of all watches in the datasets that is represented by this cluster.

0.76	watch
0.25	yin_yang
0.03 - 0.58	octopus - starfish
0.05 - 0.45 - 0.02 - 0.25 - 0.44 - 0.29 - 0.03 - 0.03 - 0.80 - 0.45	anchor - ant - ceiling_fan - crab - crayfish - lobster - mayfly - octopus - scorpion - tick
0.99 - 0.99	Faces - Faces_easy
0.09	Motorbikes
0.35 - 0.93	chair - windsor_chair
0.87 - 0.23 - 0.27	Motorbikes - inline_skate - wheelchair
0.86 - 0.86	ketch - schooner
0.69	hawksbill
0.82 - 0.03	dalmatian - panda
0.21 - 0.78 - 0.03	cougar_body - cougar_face - wild_cat
0.05 - 0.86 - 0.08	lotus - sunflower - water_lilly
0.01	butterfly
0.11	cougar_body
0.40	brain
0.82 - 0.56	Leopards - wild_cat
0.52	hedgehog
0.31	airplanes
0.62 - 0.11	airplanes - helicopter
0.41	tick
0.02 - 0.61 - 0.01 - 0.66	brontosaurus - elephant - llama - rhino
0.64	accordion
0.81	grand_piano
0.84	trilobite
0.62 - 0.18 - 0.03 - 0.14 - 0.83 - 0.15 - 0.02	emu - gerenuk - ibis - kangaroo - llama - okapi - rooster
0.71	revolver
0.34	kangaroo
0.15	nautilus
0.64 - 0.31 - 0.61	flamingo - flamingo_head - ibis
0.07 - 0.04 - 0.68 - 0.03 - 0.02	buddha - cup - ewer - lamp - menorah
0.31	soccer_ball
0.31	nautilus
0.58 - 0.02	soccer_ball - yin_yang
0.03	butterfly
0.05	butterfly
0.88	laptop
0.39	stop_sign
0.62 - 0.03	euphonium - saxophone

it is likely that some of them are more specific object categories compared to the Caltech101 classes. A thorough analysis of the labels differences could probably yield more insight into the Convnet's performance on Caltech101 but this would be a cumbersome task compared with an automatic clustering scheme. The proposed method therefore provides a useful tool to analyze the learned context across datasets.

Fig. 5: Clustering the vector representation of all images in Caltech101. The feature vector representation was extracted from layer 14 in VGGNet. Every column represents a separate cluster and shows four random images from each.

5 Conclusion

In this paper we have proposed a technique for clustering and visualizing internal representations in a pretrained Convnet, based on Dirichlet Process Gaussian Mixture Models. Our clustering approach is unsupervised, which makes it possible to interpret relevant information from data representation regardless whether labels are available as oppose to related approaches that visualize all nodes. The method copes with the high number of nodes in a single layer of a Convnet by clustering them to a discrete label space. This reveals how many feature clusters a layer uses to represent a given image or set of images, and whether this is a meaningful representation can be interpreted by the user.

The proposed algorithm is well suited to explore and explain the cross domain generalizability that has been experimentally shown in Transfer Learning research recently. Our experiments showed that a Convnet trained on a dataset can have meaningful representations of unseen data, despite the new data being very dissimilar to the original one. Interestingly, we found that even these meaningful representations are not limited to shallow layers which often are considered very basic feature descriptors. Even high layers contained representations that were useful on a very dissimilar dataset. One future extension of this method could be to combine it with the deconvolutional method presented in [23], and do the clustering on deconvolved representations of the nodes in each layer. This would yield a higher spatial size of the label maps and allow for fully connected layers to be presented in label maps as well as for the convolutional layers.

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dan Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. Software available from tensorflow.org.
2. James Bergstra, Olivier Breuleux, Frédéric Bastien, Pascal Lamblin, Razvan Pascanu, Guillaume Desjardins, Joseph Turian, David Warde-Farley, and Yoshua Bengio. Theano: a CPU and GPU math expression compiler. In *Proceedings of the Python for Scientific Computing Conference (SciPy)*, June 2010. Oral Presentation.
3. David M. Blei, Michael I. Jordan, et al. Variational inference for Dirichlet process mixtures. *Bayesian analysis*, 1(1):121–144, 2006.

4. Tianhorng Chang and C-CJ Kuo. Texture analysis and classification with tree-structured wavelet transform. *IEEE transactions on image processing*, 2(4):429–441, 1993.
5. Jia Deng, Wei Dong, Richard Socher, Li jia Li, Kai Li, and Li Fei-fei. Imagenet: A large-scale hierarchical image database. In *CVPR*, 2009.
6. Martin Ester, Hans-Peter Kriegel, Jörg Sander, Xiaowei Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the 2nd International Conference on Knowledge Discovery and Data Mining*, volume 96, pages 226–231. AAAI Press, 1996.
7. Li Fei-Fei, Rob Fergus, and Pietro Perona. One-shot learning of object categories. *IEEE transactions on pattern analysis and machine intelligence*, 28(4):594–611, 2006.
8. Li Fei-Fei, Rob Fergus, and Pietro Perona. Learning generative visual models from few training examples: An incremental bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding*, 106(1):59–70, 2007.
9. Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. Imagenet classification with deep convolutional neural networks. In *Advances in neural information processing systems*, pages 1097–1105, 2012.
10. Yann LeCun, Fu Jie Huang, and Leon Bottou. Learning methods for generic object recognition with invariance to pose and lighting. In *Computer Vision and Pattern Recognition, 2004. CVPR 2004. Proceedings of the 2004 IEEE Computer Society Conference on*, volume 2, pages II–97. IEEE, 2004.
11. Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3431–3440, 2015.
12. Aravindh Mahendran and Andrea Vedaldi. Understanding deep image representations by inverting them. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5188–5196, 2015.
13. Bangalore S Manjunath and Wei-Ying Ma. Texture features for browsing and retrieval of image data. *IEEE Transactions on pattern analysis and machine intelligence*, 18(8):837–842, 1996.
14. Rickard Norlander, Josef Grahn, and Atsuto Maki. *Wooden Knot Detection Using ConvNet Transfer Learning*, pages 263–274. Springer International Publishing, 2015.
15. Jayaram Sethuraman. A constructive definition of dirichlet priors. *Statistica sinica*, pages 639–650, 1994.
16. Jamie Sherrah. Fully convolutional networks for dense semantic labelling of high-resolution aerial imagery. *arXiv preprint arXiv:1606.02585*, 2016.
17. Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.
18. Korsuk Sirinukunwattana, Josien PW Pluim, Hao Chen, Xiaojuan Qi, Pheng-Ann Heng, Yun Bo Guo, Li Yang Wang, Bogdan J Matuszewski, Elia Bruni, Urko Sanchez, et al. Gland segmentation in colon histology images: The glas challenge contest. *arXiv preprint arXiv:1603.00275*, 2016.
19. Korsuk Sirinukunwattana, David R.J. Snead, and Nasir M. Rajpoot. A stochastic polygons model for glandular structures in colon histology images. *IEEE transactions on medical imaging*, 34(11):2366–2378, 2015.
20. Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In *Advances in neural information processing systems*, pages 3320–3328, 2014.
21. Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? In Z. Ghahramani, M. Welling, C. Cortes, N. D. Lawrence, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 27*, pages 3320–3328. Curran Associates, Inc., 2014.
22. Jason Yosinski, Jeff Clune, Anh Nguyen, Thomas Fuchs, and Hod Lipson. Understanding neural networks through deep visualization. *arXiv preprint arXiv:1506.06579*, 2015.
23. Matthew D. Zeiler and Rob Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer, 2014.